Université de Toulouse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

L'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue par :

Jim MAINPRICE

Le 17 décembre 2012

Titre :

Planification de mouvement pour la manipulation d'objets sous contraintes d'interaction homme-robot

École doctorale et discipline ou spécialité :

EDSYS : Robotique et Informatique

Unité de recherche :

Laboratoire d'Analyse et d'Architecture des Systèmes

Directeur(s) de Thèse :

M. Thierry SIMÉON

Rapporteurs :

Mme. Véronique PERDEREAU
M. Thierry FRAICHARD

Autre(s) membre(s) du jury :

Mme. Marilena VENDITTELLI
M. Daniel SIDOBRE
M. Rachid ALAMI

Université de Toulouse
École Doctorale Systèmes

Thèse

en vue de l'obtention du

Doctorat de l'Université de Toulouse
délivré par l'Institut National des Sciences Appliquées de Toulouse

Spécialité: Robotique et Informatique

présentée et soutenue publiquement le 17 décembre 2012

Planification de mouvement pour la manipulation d'objets sous contraintes d'interaction homme-robot

Jim MAINPRICE

Préparée au Laboratoire d'Analyse et d'Architecture des Systèmes
sous la direction de M. Thierry SIMÉON

Jury

Mme. Véronique PERDEREAU	Rapporteur
M. Thierry FRAICHARD	Rapporteur
Mme. Marilena VENDITTELLI	Examinateur
M. Rachid ALAMI	Examinateur
M. Daniel SIDOBRE	Examinateur
M. Thierry SIMÉON	Directeur de Thèse

À mes parents

Remerciments

Je tiens à remercier Rachid Alami, directeur du groupe RIS, pour m'avoir fait confiance dès le départ, m'avoir permis de terminer cette thèse dans de bonnes conditions et avoir accepté de présider mon jury. Je lui suis aussi très particulièrement reconnaissant pour son immense soutien lors de notre déplacement à Boston. Un grand merci va également à mon directeur de thèse Thierry Siméon qui a toujours fait preuve de patience et de rigueur, notamment au moment de la rédaction de ce manuscrit. Ensuite je voudrais adresser de sincères remerciements à Daniel Sidobre qui a conduit le projet Dexmart au LAAS, projet qui a financé la plus grande partie de mes travaux de thèse. IL a été très disponible et agréable et j'ai participé avec lui à de nombreuses collaborations européennes très enrichissantes. Je le remercie d'avoir accepter d'être examinateur de ma thèse. Je tiens aussi à remercier mes deux rapporteurs, Véronique Perdereau de l'ISIR et Thierry Fraichard de l'INIRIA Grenoble pour leur lecture minutieuse du manuscrit et leur rapports très éclairants. Merci également à Marilena Vendittelli pour avoir accepté d'être examinatrice de ma thèse.

Au LAAS, j'ai eu le plaisir de côtoyer des personnes d'une grande qualité scientifique et humaine. Je pense tout d'abord à Romain Iehl avec qui j'ai partagé le bureau et qui m'a beaucoup aidé et appris. Je n'oublirai pas non plus Léonard Jaillet, Akin Sisbot, Mokhtar Gharbi et Xavier Borquère dont l' aide m'a été très précieuse pour démarrer et à Jean-Philipe Saut, Mamou Gharbi et Séverin Lemaignan avec qui j'ai travaillé par la suite. Une pensée va également à ceux qui ont été un réel soutien dans les moments difficiles comme Matthieu Warnier, Wassim Filali ou Yi Li. Je remercie Matthieu Herrb et Anthony Mallet pour leur expertise m'ayant permis d'éviter les pièges insondables de Linux. Je tiens tout particulièrement à remercier Arnaud Degroote, conseiller technique et scientifique infaillible et compagnon de voyage.

Finalement, je remercie ma famille qui a été présente pendant ces trois années au téléphone et souvent en coulisses. Sans eux cette thèse ne serait jamais arrivée à son terme dans le délai imparti. Grand merci aussi à mes amis, Thomas, Rémy, les trois Olivier, Florian, Guillaume, Tigrane, Virginie, Eugénie, Anabel, Pauline et Eve qui ont tous été présents dans les moments de doutes.

Résumé

Un robot agit sur son environnement par le mouvement, sa capacité à planifier ses mouvements est donc une composante essentielle de son autonomie. Dans ce contexte, l'objectif de cette thèse est de concevoir des méthodes algorithmiques performantes permettant le calcul automatique de trajectoires pour des systèmes robotiques complexes dans le cadre de la robotique d'assistance. Les systèmes considérés qui ont pour vocation de servir l'homme et de l'accompagner dans des tâches du quotidien doivent tenir compte de la sécurité et du bien-être de l'homme. Pour cela, les mouvements du robot doivent être générés en considérant explicitement le partenaire humain raisonant sur un modèle du comportement social de l'homme, de ses capacités et de ses limites afin de produire un comportement synergique optimal.

Dans cette thèse nous étendons les travaux pionniers menés au LAAS dans ce domaine afin de produire des mouvements considérant l'homme de manière explicite dans des environnements encombrés. Des algorithmes d'exploration de l'espace des configurations par échantillonnage aléatoire sont combinés à des algorithmes d'optimisation de trajectoire afin de produire des mouvements sûrs et agréables. Nous proposons dans un deuxième temps un planificateur de tâche d'échange d'objet prenant en compte la mobilité du receveur humain permettant ainsi de partager l'effort lors du transfert. La pertinence de cette approche a été étudiée dans une étude utilisateur. Finalement, nous présentons une architecture logicielle qui permet de prendre en compte l'homme de manière dynamique lors de la réalisation de tâches de manipulation interactive.

Abstract

A robot acts upon its environment through motion, the ability to plan its movements is therefore an essential component of its autonomy. Motion planning is an area of research that has been widely studied in recent decades. The objective of this thesis is to design algorithmic methods to perform automatic trajectory computation for complex robotic systems in the context of assistive robotics. This emerging field of autonomous robotics applications brings new constraints and new challenges. Such systems that are designed to serve humans and to help in daily tasks must consider the safety and well-being of the surrounding humans. To do this, the robot's motions must be generated by considering the human partner explicitly. For comfort and efficiency, the robot must take into account a model of human social behavior, capabilities and limitations to produce an optimal synergistic behavior.

In this thesis we extend to cluttered environments the pioneering work that has been conducted at LAAS in this field. Algorithms that explore the configuration space by random sampling are combined with trajectory optimization algorithms to produce safe and human aware motions. Secondly we propose a planner for object handover taking into account the mobility of the human recipient allowing to share the effort during the transfer. The relevance of this approach has been studied in a user study. Finally, we present a software architecture that allows to take dynamically into account humans during the execution of interactive manipulation tasks. This architecture, developed in collaboration with a partner of the European project Dexmart was also evaluated in a user study.

Table des matières

1

Introduction

Le terme *robotique* a été introduit par *Isaac Assimov* dans un récit de science fiction publié en 1941. Ce terme désigne la discipline s'intéressant au fonctionnement et à l'utilisation des robots pour effectuer des tâches de manière automatique. De nos jours, l'objectif de la robotique dépasse la mise en oeuvre de tâche simples dans des environnements structurés. Il vise le développement de systèmes robotiques capables d'évoluer dans des environnements variés, connus ou non, afin de réaliser des tâches de plus en plus complexes, de la manière la plus autonome possible. Les applications vont de l'exploration spatiale à la chirurgie, en passant par l'aide à la personne.

Pour réaliser des tâches dans un monde physique, un système robotique agit sur son environnement par le mouvement. Sa capacité à planifier ses mouvements est donc une composante essentielle de son autonomie.

La planification de mouvement est un domaine de recherche qui tend à développer des méthodes algorithmiques permettant de calculer de manière automatique des trajectoires pour des systèmes complexes. Ce domaine, largement étudié ces dernières années [Latombe 91a, LaValle 06], a donné lieu à l'émergence de techniques probabilistes [Kavraki 96, Švestka 97]. Celles-ci sont largement utilisées dû à leur efficacité pour traiter des problèmes complexes. Leur champ d'application s'est notamment élargi à la conception assistée par ordinateur, à l'animation graphique et à la biologie moléculaire. En outre, la robotique d'assistance à la personne qui est un des champs d'application émergeant de la robotique autonome, apporte de nouvelles contraintes et de nouveaux défis à la robotique. De tel systèmes qui ont pour vocation de servir l'homme et de l'accompagner dans des tâches du quotidien doivent tenir compte de la sécurité et du bien-être de l'homme. Ceci sera l'objet de cette thèse. En effet, l'homme ne peut être considéré comme un simple obstacle pour le robot. En plus d'être capable de se mouvoir dans l'environnent de

manière autonome, l'homme est un être social dont le partage de l'espace est régi par une ensemble de normes [Hall 66]. Pour plus de confort et d'efficacité le robot doit tenir compte d'un modèle du comportement social de l'homme, de ses capacités et de ses limites afin de produire un comportement synergique optimal. Une première approche au partage de l'espace intelligent à été introduite dans [Sisbot 08b] mais les techniques développées se limitent à des systèmes simples et des environnements peu encombrés.

1.1 Contributions de la thèse

Le travail présenté dans ce manuscrit porte sur le problème de planification de mouvement pour des torses humanoïdes dans le but de réaliser des tâches de navigation et de manipulation en présence de l'homme mais aussi de manière conjointe avec ce dernier.

Nous proposons, en premier lieu, un planificateur probabiliste capable de traiter des tâches d'échange d'objet et de déplacement dans des environnements encombrés. Ce planificateur tient compte de contraintes de sécurité mais aussi de confort issu de la théorie "proxemique". Il se décompose en deux étapes. Premièrement, l'exploration générale d'une carte de coût de haute dimension. Cette carte de coût résulte de la modélisation des contraintes sous forme d'une fonction de coût dans l'espace des configurations du robot. La deuxième étape consiste en une phase de lissage et d'optimisation locale de la solution au voisinage de la trajectoire résultante de la phase initiale.

Deuxièmement est une méthode algorithmique pour le calcul de trajectoire d'échange d'objet entre le robot et l'humain, permettant de partager l'effort. Elle dote le robot de capacités proactives. Il robot peut proposer des solutions d'échange d'objet intelligentes, par dessus une table ou à travers une fenêtre. La méthode procède en échantillonnant des positions d'échange d'objet dans l'espace des configurations composé du produit des deux sous espaces respectif de l'homme et du robot. Les configurations sont évaluées aux regard des critères associés au confort et à la mobilité de l'homme. La configuration d'échange qui satisfait le mieux une combinaison des différents critères est ensuite calculée par l'algorithme. Cette méthode a été évaluée auprès d'une trentaine de sujets.

Enfin la dernière contribution, nous avons développé un système pour la manipulation interactive permettant de prendre en compte l'homme de manière dynamique lors de la manipulation d'objet. Ce système est basé sur la formalisation des contraintes HRI, mais également sur les algorithmes de planification de mouvement, de génération de trajectoire et d'asservissement, ainsi que sur un système attentionel bio-inspiré. Ce système permet d'allouer de manière dynamique une tâche de manipulation au robot telle que saisir, poser, recevoir ou donner un objet et de l'exécuter en prenant en compte l'humain de manière explicite. Ce système a été intégré et évalué auprès de cinq sujets.

1.2 Organisation du manuscrit

Tout d'abord nous établirons un état de l'art des grands principes et des bases de la planification de mouvement et des algorithmes détaillés dans cette thèse (Chapitre 2), suivi d'un état de l'art sur la prise en compte de l'homme explicite en planification de mouvement et de l'échange d'objet homme-robot. Nous détaillerons par la suite les algorithmiques mis en oeuvre pour prendre en compte l'homme tout en planifiant des chemins dans des espaces de haute dimension (Chapitre 3). Nous décrirons ensuite un planificateur d'échange d'objet homme-robot prenant en compte la mobilité de l'homme (Chapitre 4). Enfin, nous présenterons une architecture logicielle qui permet de prendre en compte l'homme de manière dynamique lors de la manipulation interactive (Chapitre 5).

1.3 Publications associées à cette thèse

Les publications suivantes sont associées à la thèse :

1. Jim Mainprice, Akin E. Sisbot, Thierry Siméon, Rachid Alami, *Planning Safe and Legible Hand-over Motions for Human-Robot Interaction*, IARP Workshop on Technical Challenges for Dependable Robots in Human Environments, Toulouse, 2010.
2. Sidobre, D., Saut, J.P., Broquere, X., Mainprice, J., *Reactive Grasping for Human-Robot Interaction*, IEEE, RSS Workshop on bimanual manipulation, Zaragoza, 2011.
3. Jim Mainprice, Akin E. Sisbot, Léonard Jaillet, Juan Cortés and Thierry Siméon, Rachid Alami, *Planning Human-aware motions using a sampling-based costmap planner*, IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 2011.
4. Sidobre, D., Broquere, X., Mainprice, J., Burattini, E., Finzi, A., Rossi, S. and Staffa, M., *Human Robot Interactions*, Advanced Bimanual Manipulation, Springer, 2012
5. Jim Mainprice, Mamoun Gharbi, Thierry Siméon, Rachid Alami, *Sharing effort in planning human-robot handover tasks*, IEEE Ro-Man, Paris, France, 2012

1.4 Contexte de la thèse

Cette thèse a été réalisée au Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS), dans le cadre des projets européens Phriends, Dexmart et Saphari. Le but de ces projets financés par l'union européenne est d'établir une collaboration entre des partenaires européens dans le domaine de la robotique de service.

Phriends (Physical Human-Robot Interaction : depENDability and Safety) concernait le développement des composants clés des prochaines générations de robots, devant respecter les standards stricts de sécurité, tout en améliorant les performances. Ceci a posé de nouveaux défis dans la conception du système, incluant la mécanique, le contrôle, les algorithmes de planification et les systèmes de supervision. Le robot Justin du DLR, respectant ces standards, a été utilisé

comme démonstrateur du projet posant ainsi le problème de planification de mouvement et de tâches de manipulation pour les torses humanoïdes.

Dexmart (DEXterous and autonomous dual-arm/hand robotic manipulation with sMART sensory-motor skills) était un projet avec l'ambition de combler le fossé entre l'utilisation de robots dans l'industrie et l'utilisation future de robots de service évoluant dans des environnements non structurés. Ce projet a contribué à l'élaboration des connaissances et techniques dans tous les domaines concernant la robotique de service qui nécessitent des tâches de manipulation dextre avec deux mains. Plusieurs défis ont été posés, notamment celui de la planification de tâches de manipulation à plusieurs bras.

Saphari (Safe and Autonomous Physical Human-Aware Robot Interaction) qui a démarré en 2012, il suit le projet Phriends, et compte opérer un changement de paradigme fondamental dans le développement des robots. Saphari place l'humain au centre de la conception. Ce projet adresse tous les aspects de l'interaction physique sure, intuitive entre les humains et des systèmes robotiques d'apparence humaine complexes et grandement interconnectés. Il place en particulier l'accent sur le robot autonome travaillant de manière collaborative avec l'homme. Il pose le problème de la prise en compte de l'homme de manière dynamique notamment dans des tâches de manipulation et d'échange d'objet.

Nous avons pris part à ces trois projets, avec plus d'intensité sur le projet Dexmart pour lequel nous avons participé à la rédaction des délivrables internes. Notre participation à ce projet à conduit à une collaboration plus étroite avec nos partenaires d'université de Naples en Italie et la rédaction en commun d'un chapitre de livre sur l'interaction homme-robot.

2

État de l'art

Un système robotique ayant une tâche de manipulation à accomplir doit être apte à se mouvoir et à transporter les objets manipulés dans un environnement contraint. La planification de mouvement est un élément essentiel dans la résolution des problèmes de manipulation auxquels sont confrontés les manipulateurs mobiles.

L'algorithmique du mouvement robotique est un thème de recherche qui a suscité de nombreuses études depuis les travaux pionniers de Lozano-Pérez [Lozano-Pérez 83]. De nombreuses approches ont été proposées [Latombe 91a, Choset 05a, LaValle 06], mais peu d'entre elles sont capables de résoudre en un temps raisonnable les problèmes complexes de manipulation mobile dans le cadre de la robotique de service. En effet, la présence de l'homme dans l'espace de travail du robot amène de nouvelles contraintes. L'étude de l'interaction homme-robot qui est un domaine actif de ces dernières années, a permis de caractériser ces contraintes. Cependant, la planification de mouvement raisonnant sur l'homme de manière explicite se concentre principalement sur des problèmes de navigation.

La première partie de ce chapitre présente un bref état de l'art des approches existantes ainsi qu'un rappel de la formulation standard des problèmes de planification de mouvement. Dans la seconde partie (Section 2.2), nous détaillons les approches basées sur des techniques probabilistes de planification de mouvement et nous introduisons des techniques récentes d'optimisation de chemin (Section 2.3). Enfin dans une dernière partie (Section 2.4), nous établissons l'état de l'art de la génération de mouvement "human-aware" et du transfert d'objet.

2.1 Planification de mouvement

Dans sa formulation de base, la planification de mouvement est connue sous le nom du problème du "*déménageur de piano*" [Schwartz 83]. Le problème de planification de mouvement s'énonce de la manière suivante : "Existe t-il un chemin libre de toute collision dans l'espace Euclidien (ou espace de travail) permettant de déplacer un ensemble de corps rigides d'une position et une orientation initiales vers une position et une orientation finales, connaissant exactement l'environnement et la position des obstacles ?". Cette section introduit des notions de base du problème (Section 2.1.1) fondamentales pour sa formulation (Section 2.1.2). Ensuite, différentes approches de planification de mouvement seront présentées dans la Section 2.1.3.

2.1.1 Notions de base

Un problème de planification de mouvement est spécifié par la description du robot et de la scène 2D ou 3D dans laquelle il se trouve ainsi que par les postures initiales et finales du robot. La description de la scène, généralement nommée *espace de travail* $\mathcal{W}$, est elle-même composée de diverses entités géométriques que l'on nomme *obstacles*. Chacune entité est composée d'un ou de plusieurs corps rigides. La position et l'orientation de ces corps sont définies par une transformation qui permet de passer d'un repère de référence de l'espace de travail $\mathcal{F}_{\mathcal{W}}$ à un repère $\mathcal{F}_{\mathcal{B}}$ associé au corps $\mathcal{B}$.

Le robot est également représenté par un ensemble de corps rigides qui sont reliés par des liaisons cinématiques. L'ensemble de ces liaisons définissent *la chaîne cinématique* du robot. Dans le cas précis d'un manipulateur, une des extrémités de la chaîne est reliée à une base fixe ou mobile, tandis que l'autre est généralement dotée d'un *organe terminal*, tel qu'un outil ou une pince, permettant la manipulation d'objets dans l'espace. Le mouvement d'un robot peut être vu comme une transformation d'une posture du robot à une autre. La posture d'un robot dans $\mathcal{W}$ peut être définie de façon unique grâce à un ensemble de paramètres nommés *degrés de liberté* définis par la chaîne cinématique du robot.

Une configuration q, définie par un vecteur multidimensionnel permet d'identifier une posture de manière unique. Chaque paramètre de ce vecteur spécifie un degré de liberté du robot. Les fonctions qui expriment la position et l'orientation de l'organe terminal en fonction d'une configuration sont appelées fonctions de *cinématique directe*. L'espace de la représentation paramétrique des postures du robot est nommé *espace des configurations* $\mathcal{C}$ du robot. L'avantage de cette représentation est que le problème de planification de mouvement se réduit à trouver un chemin sans collision pour un point représentant la configuration du robot. Cette simplification induit une dimension de l'espace des configurations égale au nombre de degrés de liberté du robot. Cette notion d'espace des configurations originaire de la mécanique a été introduite en robotique par Lozano-Pérez [Lozano-Pérez 83]. La portion de $\mathcal{C}$ décrivant l'ensemble des configurations sans collisions est notée $\mathcal{C}_{free}$. De même l'ensemble des configurations en collision est notée $\mathcal{C}_{obst}$.

2.1.2 Formulation du problème

Etant donnée q_{init} la configuration initiale et q_{final} la configuration finale, nous pouvons définir le problème de planification de mouvement comme la détermination d'une application continue $\tau : [0,1] \rightarrow \mathcal{C}$, avec $\tau(0) = q_{init}$ et $\tau(1) = q_{final}$ de telle sorte qu'aucune configuration sur le chemin ne soit en collision avec les obstacles. Le chemin solution τ est dit *faisable*, si il est *libre* de collision, et *admissible* si il respecte les différentes contraintes internes du robot (*e.g.* contraintes de chaîne fermée, contraintes différentielles,...).

Formulé de cette façon, le problème de planification de mouvement consiste en l'exploration de la connectivité du sous-espace $\mathcal{C}_{free}$. Une solution existe si et seulement si q_{init} et q_{final} appartiennent à la même composante connexe de ce sous-espace. La principale difficulté de cette formulation est la représentation du sous-espace $\mathcal{C}_{obst}$. Nous allons voir dans la Section 2.1.3 et la Section 2.2 des méthodes qui s'affranchissent de cette représentation en utilisant les modèles 3D du robot et des obstacles combinés à des algorithmes de détection de collision. Les documents de référence dans le domaine [Canny 88, Latombe 91a, Choset 05a, LaValle 06] donnent plus de détails sur cette formulation du problème de planification de mouvement.

2.1.3 Méthodes de planification

L'effort de recherche des dernières décennies a abouti à trois types d'approches développées de manière parallèle : les méthodes de décomposition cellulaire [Schwartz 83], des champs de potentiel [Khatib 86] et de construction de graphes [Nilsson 69]. Dans ces trois familles de méthodes, on distingue des planificateurs *complets* (méthodes exactes), *complets en résolution* ou *en probabilité*.

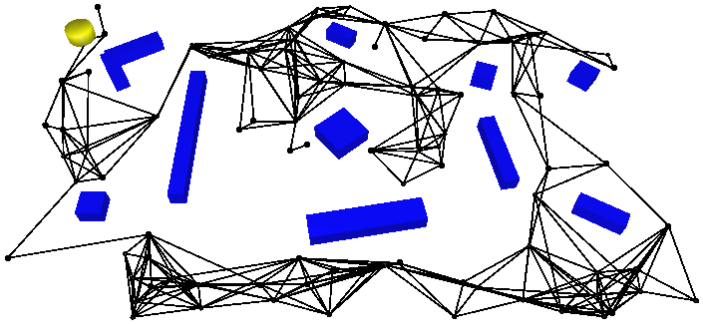
Les méthodes à champs de potentiel sont très efficaces pour traiter des problèmes d'évitement d'obstacles de manière réactive [Khatib 86, Koren 91]. Elles consistent à spécifier la configuration but comme attracteur et les obstacles comme répulseur définissant ainsi un champ de potentiel sur l'espace des configurations. La planification est résolue par descente du gradient afin d'atteindre le pôle attracteur. Ces méthodes sont limitées à traiter des problèmes simples car la recherche peut se trouver bloquée dans un minimum local du champ de potentiel. Les méthodes exactes [Schwartz 83, Canny 88] garantissent de trouver un chemin solution s'il existe, ou de signaler qu'aucune solution n'est possible. Cependant ces méthodes n'ont qu'un intérêt théorique car elles ne permettent pas de résoudre des problèmes de planification pratiques dans des temps raisonnables. En effet, il a été montré que le problème est intrinsèquement difficile [Reif 79] et que sa complexité croît exponentiellement avec le nombre de degrés de liberté du robot [Canny 88]. D'autres approches basées sur la discrétisation de $\mathcal{C}$ ont été proposées [Faverjon 84, Lozano-Pérez 87]. Ces planificateurs garantissent de trouver un chemin solution s'il existe, mais uniquement à une résolution donnée ; ils sont donc complets en résolution.

Dans les années 90, Barraquand et al. ont introduit l'échantillonnage aléatoire afin de surmonter les limitations des méthodes de champs de potentiel dans [Barraquand 91]. Des planificateurs plus généraux sont ensuite apparus ne reposant plus sur une construction préalable

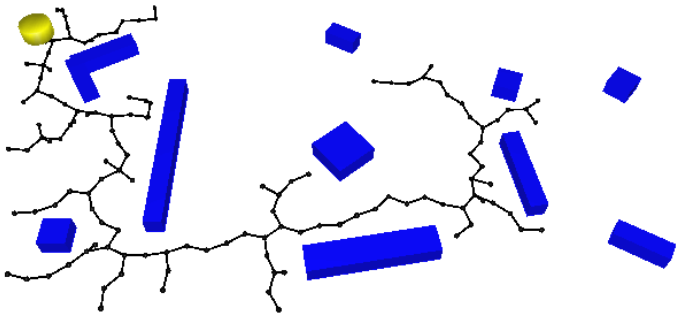
d'une représentation exacte ou approchée de l'espace des configurations [Kavraki 96, Švestka 97, LaValle 98]. Ces méthodes de planification, dites *méthodes probabilistes*, permettent de faire face à la complexité inhérente au problème. Le terme *probabiliste* caractérise la complétude probabiliste de ces algorithmes qui garantissent de trouver une solution en un temps fini si elle existe. Ces méthodes, très efficaces en pratique, sont aujourd'hui largement utilisées. La section suivante fait la synthèse de travaux récents sur ces techniques.

2.2 Planification par échantillonnage aléatoire

La planification de mouvement s'est longtemps heurtée à la complexité combinatoire inhérente au problème. Les approches probabilistes apparues depuis une quinzaine d'années permettent aujourd'hui de résoudre de nombreux problèmes qui restent hors de portée des méthodes déterministes initialement développées. C'est le cas de l'exploration des espaces de recherche hautement dimensionnés. Le champ d'application de ces approches dépasse aujourd'hui la seule robotique. Elles sont utilisées dans des domaines aussi diversifiés que la CAO, l'animation graphique ou la biologie moléculaire.



(a) Réseau Probabiliste



(b) Diffusion

FIGURE 2.1 – Les deux grandes familles de méthodes probabilistes : les méthodes de construction de réseaux probabilistes (a) et les méthodes de diffusion (b) appliquées à un robot cylindrique évoluant dans le plan.

Les algorithmes probabilistes consistent en la construction d'un graphe capturant la connexité de l'espace libre $\mathcal{C}_{free}$ en utilisant l'aléa, dans le but de s'affranchir d'une représentation explicite de l'espace de configuration $\mathcal{C}$. On distingue deux grandes familles de méthodes probabilistes : les méthodes de construction de *réseaux probabilistes* aussi appelées PRM [Kavraki 96, Švestka 97], et les méthodes de *diffusion* ou *incrémentales* (*e.g.* RRT [LaValle 98], EST [Hsu 97], ...). Le choix de l'utilisation d'une méthode ou de l'autre dépend fortement de l'application. La Figure 2.1 montre un graphe produit par deux algorithmes de ces familles de méthodes sur un exemple de navigation où le robot est un cylindre. Dû au caractère probabiliste de ces deux familles de méthodes, une phase d'optimisation est exécutée de manière à produire des chemins lisses et réguliers.

2.2.1 Les réseaux probabilistes

Les réseaux probabilistes conviennent aux applications où plusieurs requêtes de planification d'un même système dans un même environnement doivent être résolues. Ils se caractérisent par une phase de pré-calcul ou d'apprentissage visant à capturer la connexité de l'espace libre dans un graphe et une phase de requêtes résolues rapidement en se connectant au réseau pré-calculé et en cherchant un chemin à travers celui-ci à l'aide d'algorithmes comme Dijkstra ou A^*. Également appelées méthodes à *requêtes multiples*. Ces méthodes ont été introduites simultanément par Kavraki et Latombe sous le nom de PRM (Probabilistic Roadmap Method) [Kavraki 95, Kavraki 96] et Švestka et Overmars sous le nom de PPP (Probabilistic Path Planner) [Švestka 97].

Phase de pré-calcul et requête

La phase de pré-calcul est décrite par l'algorithme 2.1. A chaque itération une configuration q est échantillonnée et intégrée au graphe en tant que noeud. Ensuite une arrête entre le nouveau noeud et le noeud considéré dans le graphe est ajoutée si un chemin sans collision existe. Les composantes connexes du graphe ainsi générées représentent des ensembles de configurations q joignables par des chemins admissibles. A l'issue de cette phase d'apprentissage, on considère qu'il n'est pas possible de joindre deux configurations n'appartenant pas à la même composante connexe. Trouver un chemin acceptable entre deux configurations revient à prouver l'appartenance de ces deux configurations à la même composante connexe. Le cas échéant, la reconstruction de la trajectoire s'effectue en trouvant dans le graphe un chemin reliant ces deux configurations puis en reconstruisant une à une les trajectoires élémentaires entre deux nœuds consécutifs.

Génération des configurations et stratégies de connexion

Plusieurs approches ont été proposées afin de palier à la difficulté de trouver les passages étroits notamment en augmentant la densité d'échantillonnage dans ces zones. En dehors des approches qui consistent à autoriser une certaine pénétration dans les obstacles [Hsu 98], il s'agit généralement d'échantillonner des configurations directement dans $\mathcal{C}_{obst}$ [Amato 98] ou

Algorithm 2.1: Algorithme PRM

```
begin
    V ← ∅, E ← ∅, n_noeuds ← 0;
    while n_noeuds < N_max do
        q ← EchantillonConf(CS);
        if q ∈ CS_free then
            V ← V ∪ q;
            n_noeuds ← n_noeuds + 1;
            V_c ← MellieursVoisins(q,V);
            pour chaque v de V_c par ordre de distance d(v,q) croissante faire
                si CheminSansCollision ((q, v)) alors
                    E ← E ∪ {(q, v)};
end
```

proches des obstacles [Boor 99]. Certaines variantes échantillonnent des configurations libres entourées de configurations en collision [Hsu 03]. Contrairement à celles-ci, certaines méthodes d'échantillonnage se concentrent sur la production de chemins de bonne qualité en échantillonnant autour de l'axe médian [Wilmarth 98]. Un intérêt a également été porté à l'échantillonnage dit *pseudo-aléatoire* [LaValle 04] afin d'obtenir une couverture plus uniforme de l'espace des configurations.

Une fois le nouvel échantillon obtenu, la stratégie de connexion la plus simple est de tenter de le connecter à tous les nœuds du graphe, ce qui génère des graphes redondants et coûteux à construire. En pratique, les connexions sont généralement limitées aux k nœuds les plus proches [Kavraki 96] ou aux nœuds situés à une distance inférieure à un seuil donné [Bohlin 00]. D'autres stratégies limitent les connexions grâce à un critère de visibilité [Nissoux 99] ou à la distance entre les deux nœuds dans le graphe [Nieuwenhuisen 04]. Dans [Jaillet 08a], le test de connexion est basé sur une notion de déformabilité ce qui permet de générer des graphes cycliques compact capturant les différentes classes d'homotopie.

Métrique dans $\mathcal{C}$ et Détection de collision

La plupart des méthodes probabilistes nécessitent de calculer la distance entre deux configurations. Une métrique idéale permettrait de prendre en compte les contraintes appliquées au système (*e.g.* les limites articulaires, les obstacles, les contraintes différentielles, ...) mais établir une telle métrique peut s'avérer extrêmement complexe. Une métrique Euclidienne normée dans $\mathcal{C}$ permet d'obtenir de bons résultats pour la planification holonome [Amato 00]. Cependant, quand le système est soumis à des contraintes différentielles, la mise en œuvre d'une métrique appropriée est plus difficile [Laumond 98, LaValle 02].

La phase la plus coûteuse (en temps de calcul) lors de la construction des réseaux probabilistes est la détection de collision des configurations et des chemins locaux les reliant. De nombreux algorithmes pour la détection de collision reposent sur une décomposition hiérarchique des objets de l'environnement ainsi que du système robotique en formes géométriques simples telles que des

sphères, des boîtes ou des enveloppes convexes [Jiménez 98, Larsen 00, Lin 03]. Par ailleurs, des méthodes comme *Lazy-PRM* [Bohlin 00] et *Fuzzy-PRM* [Nielsen 00], retardent l'appel au détecteur de collision au maximum de façon à vérifier uniquement les nœuds et les arêtes utilisées dans le chemin final calculé.

Optimalité des solutions

La solution optimale à un problème de planification de mouvement dépend du critère que l'on mesure. Typiquement, on cherche le chemin sans collision le plus court allant de la configuration initiale à la configuration finale. La version standard PRM ne converge pas vers un chemin optimal. Cette propriété a été démontrée par Karaman et Frazzoli dans [Karaman 11]. Ils ont également montré que la version simplifiée de PRM appelée sPRM [Kavraki 98] qui permet une connexion des échantillons dans leur composante connexe converge vers une solution optimale. De plus Karaman et Frazzoli ont introduit un algorithme PRM$*$ permettant de converger efficacement vers l'optimal. Dans l'algorithme standard PRM, ainsi que dans les versions modifiées sPRM et PRM$*$ les tentatives de connections entre le nouvel échantillon et les noeuds du graphe se font dans une boule de rayon fixe. Cette constante r est donc un paramètre de PRM. L'algorithme PRM$*$ est similaire à sPRM, avec la différence que le rayon de connexion r est choisit en fonction du nombre de nœuds dans le graphe n , i.e., $r = r(n) := \gamma_{PRM}(log(n)/n)^{1/d}$, où $\gamma_{PRM} > \gamma^*_{PRM} = 2(1 + 1/d)^{1/d}(\mu(C_{free})/\zeta_d)^{1/d}$, d est la dimension de l'espace C , $\mu(X_{free})$ décrit la mesure de Lebesgue (i.e. le volume) de l'espace libre, et ζ_d est le volume de la boule unitaire dans le espace Euclidien de dimension d. Le rayon r de connexion décroit avec le nombre d'échantillons. La vitesse de décroissance est telle que le nombre moyen de tentatives de connexions du noeud du graphe est proportionnel à $log(n)$.

2.2.2 Les méthodes de diffusion

Les méthodes de diffusion, construisent de manière incrémentale des arbres de recherche (*i.e.* des graphes ne contenant aucun cycle) à partir des configurations initiales et finales mais ne cherchent pas, contrairement aux précédentes, à calculer toute la connexité de $\mathcal{C}_{free}$. Leur performance vient du fait qu'elles ne nécessitent pas de phase de pré-calcul. Ces méthodes sont adaptées pour la résolution de problèmes de désassemblage [Chang 95], où les configurations initiales et/ou finales sont dans des zones très contraintes. On distingue généralement des versions monodirectionnelles qui ne développent qu'un seul arbre et des versions bidirectionnelles où les deux arbres générés ont respectivement pour racine la configuration initiale et la configuration finale du problème.

Premières approches

La première approche de diffusion fut développée par [Barraquand 91] afin de permettre à l'algorithme, basé sur une méthode de champ de potentiel, de sortir des minima locaux. Cependant ce planificateur nécessite le réglage de nombreux paramètres qui agissent directement sur

la performance. La technique proposée dans [Hsu 97] utilise une stratégie de diffusion basée sur la densité d'échantillonnage. Le nœud à étendre est choisi aléatoirement avec une probabilité inversement proportionnelle à la densité du graphe (nombre de nœuds situés dans un voisinage). La méthode d'exploration RRT (Rapidly-exploring Random Tree) [LaValle 98], est devenue la méthode de diffusion la plus populaire ces dernières années. La différence avec la méthode proposée [Hsu 97] réside dans la sélection de la direction d'expansion de l'arbre qui est déterminée le noeud à étendre. Ce principe permet une croissance rapide de l'arbre vers les grandes régions inexplorées de l'espace des configurations (voir Figure 2.2). Cet algorithme a été initialement développé pour résoudre des problèmes de planification sous contraintes kinodynamiques puis adapté aux cas simples des systèmes holonomes qui peuvent être directement formulés dans $\mathcal{C}$ [Kuffner 00, LaValle 01b]. Comme pour les méthodes citées dans le paragraphe précédent, les algorithmes RRT se composent d'une phase de construction et d'une phase de requête.

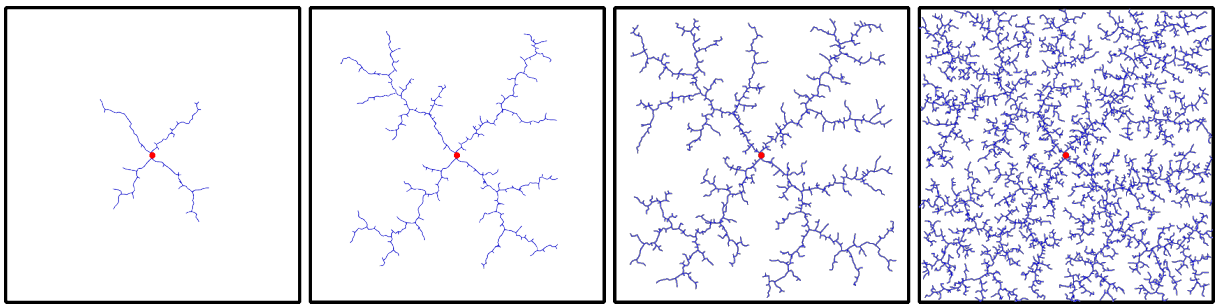


FIGURE 2.2 – Évolution d'un arbre couvrant l'espace libre par la méthode RRT

L'algorithm RRT

Les arbres RRT sont calculés suivant l'algorithme 2.2. La méthode consiste à échantillonner aléatoirement une configuration q_{rand}. La configuration ne doit pas être nécessairement valide car elle ne sera pas obligatoirement incluse dans le graphe. La droite passant par cette configuration et le nœud de l'arbre le plus proche q_{near} détermine le nœud et la direction d'extension de l'arbre.

Algorithm 2.2: Algorithme du RRT

begin
  $V \leftarrow V \cup q_{init}$;
  while pas `ConditionArret`(V, q_{goal}) **do**
    $q_{rand} \leftarrow$ `EchantilloneConf`(C) ;
    $q_{near} \leftarrow$ `MeilleurVoisin`(V, C) ;
    $q_{new} \leftarrow$ `ExtensionArbre`(q_{rand}, q_{near});
    if pas `Similaire`(q_{near}, q_{rand}) **then**
      $V \leftarrow V \cup q_{new}$;
      $E \leftarrow E \cup \{(q_{near}, q_{new})\}$;
end

L'extension des arbres se fait en deux temps. Un tirage aléatoire permet de déterminer le nœud de l'arbre que l'on tentera d'étendre (choix du plus proche voisin) . La configuration tirée n'est pas nécessairement contenue dans l'espace libre. Par contre, elle donne la direction d'extension de l'arbre. Ensuite, on recherche une nouvelle configuration réalisable (appartenant à l'espace libre) sur le chemin local défini par cette direction. Il y a deux manières de placer cette nouvelle configuration. Dans la méthode RRT basique, plus communément nommée "RRT-Extend", l'extension se fait avec un pas régulier, voir Figure 2.3. Ceci signifie que les configurations générées sont au plus à une distance ε du nœud de l'arbre le plus proche. La variante "RRT-Connect" proposée dans [Kuffner 00] réitère la phase d'extension de RRT-Extend tant que les contraintes du système (*e.g.* absence de collisions) sont respectées.

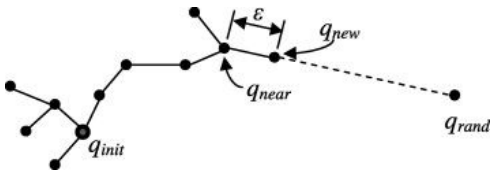


FIGURE 2.3 – RRT-Extend développe l'arbre par pas réguliers de q_{near} vers q_{rand}

Après avoir vérifié qu'il existe un chemin local entre l'arbre et la nouvelle configuration, on ajoute celle-ci à l'arbre ainsi que le segment qui relie ces configurations. L'algorithme se termine lorsque l'on peut rejoindre l'objectif à partir de l'arbre initial ou lorsque l'on a dépassé le nombre maximum autorisé de nœuds . L'intérêt de ces algorithmes vient d'un guidage efficace de l'exploration : la probabilité pour un nœud d'être sélectionné pour l'extension est proportionnelle à sa région de Voronoï.

Variantes de RRT

De manière similaire à PRM de nombreuses extensions de l'algorithme ont été proposées. Des variantes de la phase d'échantillonnage ont été introduites afin de faciliter la recherche de passages étroits dans [Rodriguez 06] ou afin de s'éloigner des zones denses de l'espace des configurations dans [Khanmohammadi 08]. La méthode proposée dans [Jaillet 05] se base sur un calcul de la région de visibilité des noeuds existant dans l'arbre de diffusion pour générer les nouveaux échantillons (voir Figure 2.4). L'adaptation du domaine d'échantillonnage permet d'améliorer les performances de l'algorithme dans des cas contraints de manière significative. Dans [Dalibard 09] une méthode basée sur l'analyse en composantes principales de la direction d'expansion de l'arbre est proposée afin d'accélérer l'exploration. Des extensions ont également été proposées afin de prendre en compte des contraintes cinématiques liées à la tâche [Berenson 09, Oriolo 09] et pour résoudre des problèmes de désassemblage [Cortés 08, Le 09].

Planification en environnements dynamiques

La planification de mouvement dans des environnements dynamiques est un problème important pour la manipulation et la navigation. Les mouvements des obstacles mobiles ne sont pas

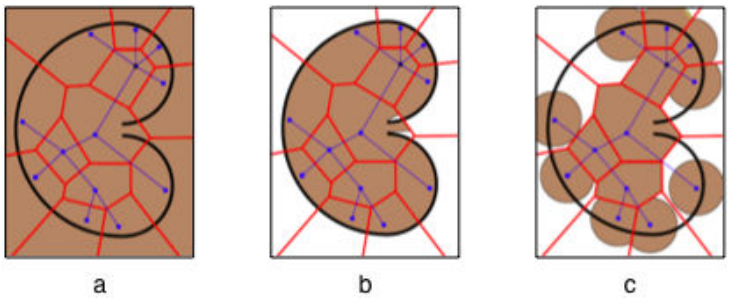


FIGURE 2.4 – Différents domaines d'échantillonnage sont montrés pour un ensemble de points dans le piège d'abeille (a) domaine RRT standard, (b) région de visibilité de Voronoi (c) domaine dynamique.

connus à l'avance et le planificateur doit raisonner sur l'extrapolation des trajectoires courantes. Les planificateurs en ligne emploient généralement un cycle continu entre la détection de l'environnement et la replanification pendant le mouvement du robot. Une notion de sécurité et d'états de collisions inévitables a été introduite [Petti 05, Bekris 07] pour modéliser le fait que l'étape longue de planification peut mettre en danger le robot.

Par ailleurs, des approches comme D∗ et D∗ lite [Stentz 95, Koenig 02] ont été développées sur la base de l'algorithme A∗. Ils appliquent des modifications minimales dans un arbre de recherche précédent afin de maintenir un chemin optimal. Cependant, ces algorithmes sont limités à des espaces de recherche de basses dimensions. C'est pourquoi des approches probabilistes ont été introduites pour traiter des environnements dynamiques dans [Leven 00, Kallman 04, Jaillet 04]. Plus récemment, des approches basées sur RRT sont apparues [Ferguson 06a, Zucker 07] basées sur un bais de l'échantillonnage et la réutilisation des parties déjà planifiées.

Planification optimale avec RRT

L'analyse de la qualité des solutions produites a reçu relativement peu d'attention. Il en est de même pour l'étude d'algorithmes adaptés à améliorer la qualité de la solution si elle est conservée au cours d'exécution. Cependant leur importance est mentionnée dans des documents séminaux comme [LaValle 01a]. L'implantation de ces algorithmes consiste souvent à calculer rapidement un chemin praticable (voir, par exemple, [Kuwata 09]). Le temps de calcul supplémentaire est consacré à l'amélioration de la solution avec des heuristiques (voir Section 2.3) jusqu'à ce que la solution soit exécutée.

Des approches ont été développées pour offrir des garanties d'optimalité. Elles sont basées sur des algorithmes de recherche classique (tels que A∗ et D∗) appliqués sur une grille qui est générée hors ligne. Récemment, ces algorithmes ont été étendus afin de fonctionner dans un mode "anytime" [Likhachev 03, Likhachev 08], avec des environnements dynamiques [Stentz 95, Likhachev 08], et pour traiter les systèmes avec des contraintes différentielles [Likhachev 09]. Leur efficacité a également été pleinement démontrée sur diverses plates-formes robotiques [Likhachev 09, Dolgov 09]. Toutefois, les garanties d'optimalité de ces algorithmes ne sont assurées que jusqu'à la résolution de grille. En outre, puisque le nombre de cellules d'une grille

augmente exponentiellement avec la dimension de l'espace des configurations, le temps d'exécution de ces algorithmes sera de même exponentiel.

Plusieurs contributions ont tenté de combler l'écart entre les méthodes probabilistes et les méthodes de grilles sur carte de coût. Dans [Urmson 03], Urmson et Simmons ont proposé des heuristiques pour biaiser la croissance des arbres RRT vers les régions qui se traduisent par des solutions peu coûteuses. Ils ont également montré des résultats expérimentaux évaluant la performance des différentes heuristiques en terme de qualité de la solution. Dans [Ferguson 06b], Ferguson et Stentz ont considéré l'exécution consécutive sur une même requête de plusieurs RRT en conservant la meilleure solution. Ils ont montré que chaque exécution de l'algorithme se traduit en moyenne par une augmentation de la qualité de la solution même si la procédure ne garantit pas de converger vers une solution optimale. Des critères pour le redémarrage de plusieurs explorations RRT ont également été proposés dans un contexte différent dans [Wedge 08]. Plus récemment Karaman et Frazzoli ont présenté dans [Karaman 11] une analyse du comportement asymptotique du coût, des solutions de différentes techniques de planification probabilistes quand le nombre d'échantillons augmente. Ils ont également introduit un algorithme basé sur RRT convergeant vers une solution optimale.

T-RRT

L'algorithme T-RRT [Jaillet 10] est un algorithme de planification permettant d'obtenir des chemins de bonne qualité dans des espaces de coût de haute dimension. Il ne s'agit pas exactement de planification optimale mais T-RRT possède de bonnes propriétés. Il converge rapidement vers des solutions de bas coût. T-RRT fait croitre un arbre qui suit les vallées et les cols de la carte de coût afin de trouver des chemins de bas coût. Un test de transition est effectué à l'ajout d'un nouveau q_{new} dans le graphe. Ce test est basé sur le critère de Métropolis communément utilisé par les méthodes d'optimisation stochastique. Le différentiel de coût entre q_{near} et q_{new}, Δc^*_{ij}, est considéré pour calculer une probabilité d'accepter la configuration dans le graphe. Cette probabilité est décrite par la formule suivante :

$$p_{ij} = \begin{cases} exp(-\frac{\Delta c^*_{ij}}{K \cdot T}) & \text{if } \Delta c^*_{ij} > 0 \\ 1 & \text{otherwise.} \end{cases} \tag{2.1}$$

où T est un paramètre auto-adaptatif de température. Quand un certain nombre de test de transitions montantes a échoué, T est automatiquement élevée et inversement pour les transitions montantes acceptées. Ce principe garantit une exploration minimale de la carte de coût. L'application du test de transition force la croissance de l'arbre à suivre les vallées de la carte de coût alors que la technique sous-jacente RRT biaise sa croissance vers les grands espaces de Voronoi de l'espace des configurations. Ces deux principes combinés permettent de planifier des chemins de bas coût dans des espaces hautement dimensionnés possiblement contraints par des obstacles de manière efficace.

RRT∗

Récemment, Karaman et Frazzioli ont proposé de nouveaux algorithmes dans [Karaman 11]. Ceux-ci sont PRM∗, présentés précédemment, ainsi que RRG et RRT∗. Ces algorithmes ont été démontrés probabilisquement complets, asymptotiquement optimaux. RRG est un algorithme incrémental qui construit un réseau probabiliste connecté offrant des performances similaires à PRM∗ dans un cadre de requête unique, et de façon "anytime" : une première solution est fournie rapidement et améliorée de façon monotone si plus de temps de calcul est disponible. L'algorithme RRT∗ est une variante de RRG qui construit progressivement un arbre. Maintenir une structure d'arbre peut être avantageux dans certaines applications, par exemple pour traiter des problèmes avec des contraintes différentielles ou pour faire face à des erreurs de modélisation.

Il a été démontré que RRT∗ converge vers une solution optimale, avec un minimum d'exigence en temps de calcul et de mémoire. RRT∗ contient deux principes qui le distinguent de la version classique de RRT. Premièrement, la connexion de nouveaux nœuds ne s'établit pas simplement au plus proche voisin mais considère les nœuds dans un voisinage du plus proche voisin afin que les feuilles de l'arbre minimisent la somme des coûts depuis la racine. Deuxièmement, des re-connexions sont considérées après l'ajout d'un noeud dans ce même voisinage car le passage par le nouveau nœud peut engendrer un chemin moins couteux de la racine aux feuilles voisines. RRT* est performant pour résoudre des problèmes de faible dimension mais converge lentement sur des problèmes de haute dimension [Iehl 12].

2.3 Optimisation de chemins et méthodes locales

Les méthodes d'optimisation, et les méthodes locales permettent d'améliorer un chemin solution et sont généralement "anytime", assurant une convergence monotone vers un minimum local. Elle procèdent par altération ou perturbation de la solution courante. Dans un premier temps, des méthodes de gradients ont été utilisées pour combler le fossé entre la planification de mouvement et le contrôle de son exécution [Quinlan 93]. Puis avec l'apparition de la planification par échantillonnage des méthodes de post-traitement ont été développées [Kavraki 98] afin d'améliorer les solutions planifiées qui sont généralement bruitées. Par ailleurs des techniques d'optimisation numériques ont également été utilisées pour produire des chemins de bonne qualité.

2.3.1 Lissage

Les chemins issus des planificateurs probabilistes sont souvent de mauvaise qualité, irréguliers et trop longs, et doivent être lissés avant leur exécution. La méthode “Shortcut” [Berchtold 94, Kavraki 98, Hsu 00] consiste à couper le chemin à lisser en trois parties et à essayer de relier leurs extrémités directement avec un chemin local unique.

Dans [Geraerts 07], une nouvelle méthode est développée qui permet de lisser des chemins en considérant les degrés de liberté séparément. Un algorithme est également proposé pour améliorer la qualité des solutions en augmentant les dégagements le long de la trajectoire mais il se limite à

des systèmes "freeflyer" simples. Dans [Guernane 09], une méthode est développée pour optimiser la longueur du chemin en le sur-échantillonnant et en utilisant une méthode de recherche pour relier les configurations éloignées.

2.3.2 Déformation et perturbation

Les méthodes par déformation sont apparues en robotique pour combler l'écart entre les méthodes de planification et les méthodes purement réactives [Quinlan 93]. Ce type de méthode est utile pour prendre en compte les données issues des capteurs et les possibles erreurs de localisation. Ces méthodes utilisent une représentation de l'espace libre autour de la trajectoire par des bulles d'espace libre dont la taille varie en fonction de la proximité des obstacles. Localement à chaque bulle, ou point étape, deux forces sont calculées : une force pour maintenir la cohésion de la trajectoire et une force pour l'évitement d'obstacle. La trajectoire est mise à jour à chaque itération en appliquant localement une combinaison des deux forces. Dans le cas du manipulateur [Brock 02], ces mises à jour sont calculées par un gradient dans l'espace des configurations. Une procédure plus complexe à été mise en oeuvre dans [Lamiraux 04] pour prendre en compte des contraintes non-holonomes. Plus récemment, dans [Delsart 08] les auteurs ont adapté ce type d'approche pour prendre en compte la dynamique dans l'espace de travail en considérant la vitesse observée des obstacles mobiles.

Plus récemment des méthodes moins réactives utilisant ce type de modélisation ont été développées pour traiter des problèmes de planification. L'algorithme est initialisé avec une trajectoire en ligne droite possiblement en collision avec les obstacles. La trajectoire est ensuite déformée afin de trouver un chemin sans collision. Ces méthodes calculent un gradient dans l'espace des trajectoires. La méthode [Ratliff 09] utilise une représentation de l'espace de travail sous forme de voxels, le robot est représenté par un ensemble de sphères. Un coût de pénétration dans les obstacles est estimé et un gradient analytique est utilisé pour produire les mises à jour. Dans [Kalakrishnan 11] le même type de représentation est utilisé mais le gradient est estimé de manière stochastique. Cette méthode permet donc de traiter des problèmes pour lesquels la fonction de coût est non linéaire ou n'est pas dérivable.

2.3.3 Optimisation numérique

L'optimisation numérique est une branche des mathématiques qui étudie les solutions algorithmiques au problème abstrait d'optimisation. Ces méthodes considèrent une formulation du problème prenant en compte des fonctions objectifs qui peuvent être linéaires ou non-linéaires. Dans [Moulard 12], elles ont été appliquées à l'optimisation de trajectoire afin de maximiser le dégagement aux obstacles à travers une interface nommée RobOptim. Cette interface procure la modélisation mathématique des fonctions objectifs et des contraintes pour utiliser des "solveurs" tel que CFSQP [Lawrence 94]. Ces techniques nécessitent le gradient voir les héssiens des fonctions objectifs et des contraintes et ne permettent généralement pas de traiter des systèmes hautement dimensionnés dans des scènes complexes dans des temps raisonnables.

2.4 Interaction Homme-Robot

Le travail de cette thèse se situe dans le contexte de la robotique de service qui se caractérise par une interaction entre l'homme et le robot. L'interaction homme-robot (HRI) est un champ d'étude consacré à la compréhension, la conception et l'évaluation des systèmes robotiques pour une utilisation par ou avec plusieurs humains. L'étude des HRI comprend différents aspects de la relation entre l'homme et le robot, allant de l'interaction cognitive à l'interaction physique. Ce chapitre ne décrit pas tout le travail sur les stratégies de contrôle pour les HRI mais se concentre sur l'interaction proche. Une étude plus générale sur les HRI est présentée dans [Goodrich 07, Kemp 07].

En effet, dans le cadre de la robotique mobile de service, le robot évolue au contact de l'homme et la sécurité ne peut pas être assurée par l'utilisation de zones interdites comme dans l'industrie. C'est pourquoi les composants matériels et logiciels doivent être conçus en tenant compte de la sécurité des personnes [Alami 06, Nonaka 04]. En plus d'assurer la sécurité grâce à une conception de robots conformes à certaines normes [Zinn 04, Bicchi 04], les mouvements du robot doivent être générés en tenant compte explicitement du partenaire humain. Dans cette section nous passons en revue les théories ainsi que les études utilisateur qui ont été menées pour déterminer les contraintes qui doivent être prises en compte pour un partage efficace de l'espace entre l'homme et le robot. Nous étudions également les différentes méthodes permettant de générer un mouvement du robot sûr et agréable.

2.4.1 Etudes utilisateurs : partage de l'espace avec le robot

Dans un contexte non interactif, la sécurité se résume à contraindre le robot à ne pas entrer en collision avec les obstacles présents dans l'espace de travail. Dans un environnement où les humains et les robots sont en interaction proche, la notion de sécurité doit être étudiée et redéfinie de manière à prendre en compte non seulement la non-collision mais également le confort de l'humain et le naturel de l'interaction. Afin de doter le robot des méthodes et des algorithmes considérant ce type de contraintes, il est important de comprendre le comportement social, à savoir quels types de mouvements sont acceptables pour les humains.

Une des principales études sur le partage de l'espace inter-humain a été menée par Hall [Hall 66]. Cette étude a introduit la théorie "proxémique", où les distances interpersonnelles sont réparties en quatre classes : intime, personnelle, sociale et publique. Elles définissent les limites spatiales de différents types d'interactions. Ces distances sont sujettes à modification en fonction des différences culturelles. Dans les cultures latines, par exemple, les distances sont plus petites alors que dans les cultures nordiques, les distances dites "proxémiques" sont plus importantes. La théorie "proxemique" divise également l'espace en trois catégories [Littlejohn 07, Lawrence-Zúniga 03]. Se sont : (1) l'espace à caractère fixe qui comprend des objets immobiles tels que les murs et les frontières territoriales ; (2) l'espace à caractère semi-fixe qui comprend des objets mobiles tels que les meubles, et (3) l'espace informel qui comprend l'espace personnel autour du corps et qui se déplace avec la personne et détermine les distances interpersonnelles.

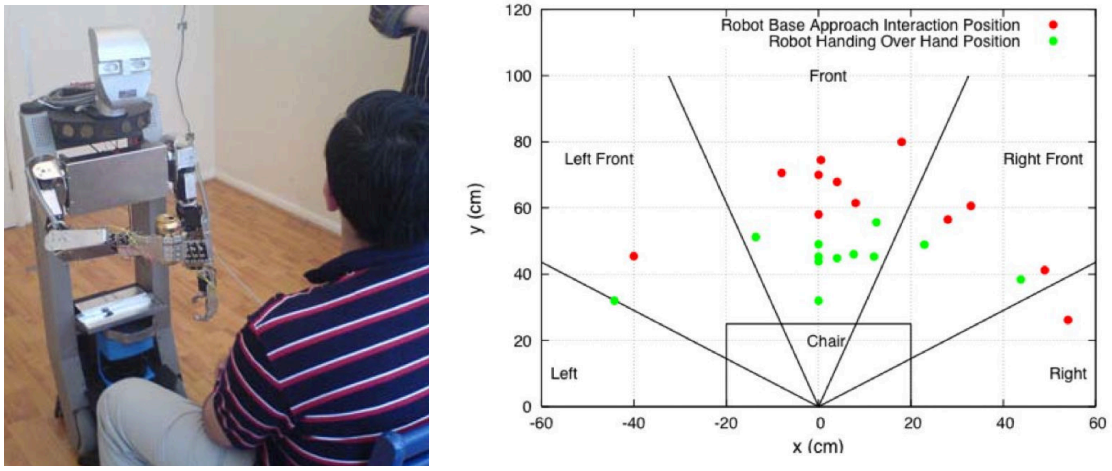
L'étude utilisateur menée par Yoda et al. [Yoda 95] prouve que dans les couloirs les humains se déplacent toujours à une vitesse constante même pendant un croisement. Dans [Pacchierotti 05, Pacchierotti 06b], Pacchierotti et al. décrivent une étude utilisateurs où les sujets marchent dans un couloir traversé par un robot dont le comportement est conforme à la théorie de la "proxémique". Le comportement du robot a été évalué en fonction de différentes vitesses du robots, distances de "signalisation" et distances "latérales". Cependant, les différences culturelles et le contexte de l'interaction ne permettent pas de généraliser les valeurs idéales trouvées.

En effet, le comportement idéal de placement spatial évalué par les sujets dépend en grande partie de la tâche et du contexte de l'interaction. Dans une étude menée par Yamaoka et al. [Yamaoka 08], les sujets sont invités à se placer dans une position confortable pour communiquer avec un robot dont la tâche est de présenter un objet. La moyenne de la distance de placement relatif au robot des sujets pour cette tâche est de 1,19 m en restant à environ 1 mètre de l'objet. Par ailleurs une autre étude, menée par Koay et al. [Koay 07] a consisté à évaluer les distances idéales d'un robot qui s'approche d'une personne pour lui remettre un objet. Les sujets ont donc été invités à décider d'une position assise confortable pour saisir une boîte apportée par le robot. La valeur résultante est une moyenne de 0,67 m de distance du robot. Le robot et les résultats de cette étude sont présentés sur la Figure 2.5. Ces deux études montrent une nette différence de placement du robot sur le confort humain en fonction de la tâche. Les distances sont plus grandes pour une tâche de communication et de présentation d'un objet où les humains sont debout que pour une tâche de remise d'objet.

Un autre aspect de l'étude de la réponse des humains aux mouvements robotiques sont les règles et les protocoles d'interaction lors de mouvements d'approche du robot. Walters et al. ont étudié cet aspect dans [Walters 05a] en se concentrant sur les distances homme-robot. Un résultat intéressant de ce travail est que le placement spatial des sujets dépend en grande partie de la façon dont ils voient le robot soit en tant qu'être social soit comme un jouet. Ce dernier point est étudié dans [Walters 05b], où une corrélation est trouvée entre les distances relatives homme-robot et les personnalités des sujets. Les résultats ont montré que plus les sujets étaient créatifs et/ou plus agressifs, plus ils étaient susceptibles de s'approcher du robot. Une étude par Dautenhahn et al. [Dautenhahn 06] a montré que les sujets préfèrent que le robot s'approche par leur gauche ou par leur droite, mais pas directement en face. Dans cette étude les sujets étaient assis et évaluaient la direction d'approche du robot dans le cadre d'une tâche de transfert d'objet. Les sujets ont évalué l'approche directe frontale moins confortable en trouvant le mouvement du robot menaçant, agressif ou s'arrêtant trop loin.

2.4.2 Etudes utilisateurs : mouvement d'un robot manipulateur

Les tâches de manipulation, comme les tâches de type chercher-et-rapporter, impliquent une interaction étroite entre l'homme et le robot. Les limites en termes de distances pour ce type de tâches sont plus courtes que les limites pour la simple navigation ou les tâches de communication. Afin de comprendre les exigences spatiales pour un scénario de manipulation, nous ne pouvons pas simplifier le placement du robot par un point dans un plan 2D, un cercle dans le cas d'un



(a) (b)

FIGURE 2.5 – Etude sur l'approche d'un robot (a) position utilisées pour le transfert d'objet (b) dans [Koay 07].

robot cylindrique, ou plus généralement, par la projection 2D de la forme du robot sur le sol.

Une étude menée par Haddadin et al. [Haddadin 08, Haddadin 07] a souligné l'importance de la vitesse du robot et de sa masse dans des scénarios d'interaction proche homme-robot. Dans ce travail, des scénarios de crash test ont été effectués entre différents robots et un mannequin à des vitesses différentes. Les résultats ont montré que le rôle de la masse (et la charge) du robot devient important si la vitesse est supérieure à 2m/s. L'importance de la dynamique est également soulignée par Huber et al. dans [Huber 08], où deux profils de vitesse différents d'un bras de robot sont comparés dans un scénario de transfert d'objet. Les sujets testés ont trouvé que le mouvement avec le profil de vitesse limité semblait plus naturel et plus confortable que le profil de vitesse trapézoïdal.

Le déplacement du bras, du haut du corps et de la tête simultanément est plus confortable qu'une séquence de ces actions. Ceci a été montré par Sakata et al. dans [Sakata 04], où ils ont évalué les mouvements d'un robot humanoïde (HRP-2). Des résultats similaires ont également été obtenus par Boekhorst et al. [Boekhorst 05] en observant le comportement d'un grand groupe d'enfants face à un robot mobile effectuant des mouvements de tête et de bras.

L'un des problèmes majeurs des études d'utilisateurs pour la manipulation est le souci de la sécurité des sujets lors du test. Dans les scénarios où le robot et les sujets sont dans une interaction étroite, tout type de mouvement du robot inattendu peut nuire, causer des blessures graves ou au moins effrayer les sujets.

2.4.3 Modélisation de l'humain

Ces études utilisateurs indiquent que dans le cas d'un robot se déplaçant dans un environnement où les humains sont présents, on doit prendre en compte certains protocoles et certaines distances sociales. Ceci implique que les humains ne peuvent pas être simplement assimilés à des

obstacles mobiles. Afin de générer des mouvements sûrs et confortables, les humains doivent être pris en compte comme des entités particulières avec des caractéristiques telles que la position et l'orientation courante, la position des mains et la capacité de manipulation. Il faut également considérer le déplacement (propriétés dynamiques), l'attention, le champ de vision de l'humain ainsi que l'activité.

Le modèle de l'humain comprend généralement sa position et son orientation, (x, y, θ) [Nakauchi 02, Takemura 07, Yoshimi 06] obtenues par la projection et l'orientation de son centre de masse. Déterminer l'orientation consiste à utiliser la direction de la poitrine comme la direction de la personne [Yoda 97]. Si la personne se déplace, on considère généralement le sens du mouvement comme la direction de la personne [Hoeller 07]. Un modèle commun pour les scénarios de manipulation est de considérer les parties importantes du corps humain. Les positions 3D de la tête, de la poitrine et des mains de la personne sont couramment utilisées afin d'avoir un modèle de la posture de l'homme [Kulić 05]. Ce modèle est également enrichi par des ellipsoïdes ayant ainsi une représentation proxémique [Pandey 09]. Puisque les humains sont généralement en déplacement dans l'environnement une représentation plus exacte est d'inclure leur vitesse à leur composante de position [Althaus 04, Martinez-Garcia 05].

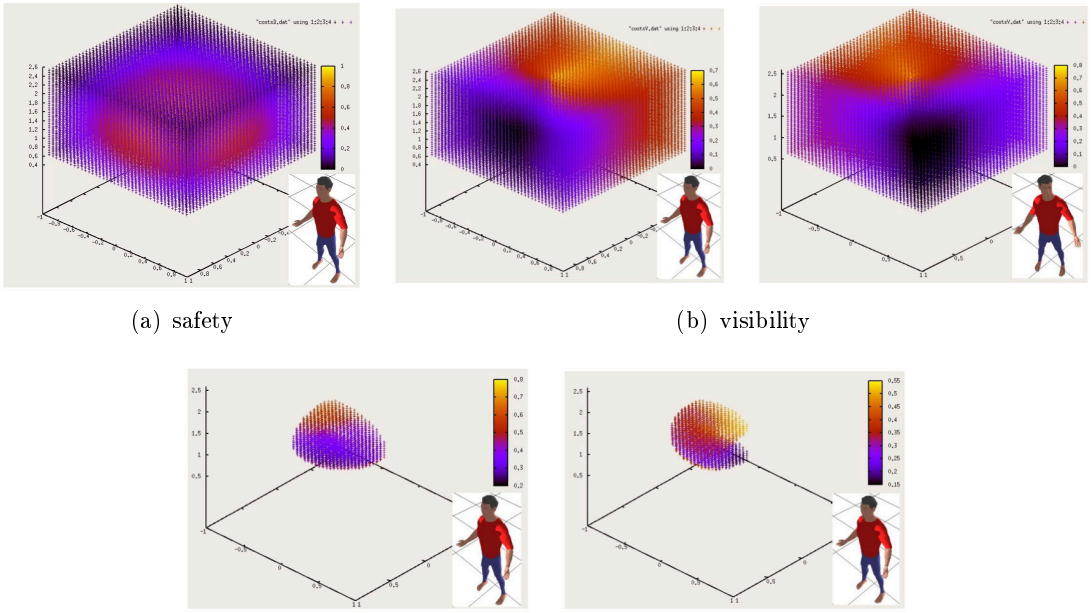
Les espaces personnels sont généralement modélisés par des seuils de distance [Hall 66, Pacchierotti 05]. L'attention est représentée par un vecteur qui caractérise la direction du regard et la région visible par un cône [Traver 00]. Des préférences supplémentaires peuvent être ajoutées au modèle de l'humain. Dans [Sisbot 07a, Sisbot 07b], les préférences telles que les fonctions gaucher et droitier sont prises en compte. Sisbot et al. utilisent également une représentation sous forme de carte de coût (voir Figure 2.6) qui permet de modéliser les notions proxémiques, de champ de vision et les capacités de manipulation. Dans [Lam 11], différents champs dans l'espace de travail sont utilisés pour modéliser l'humain et le robot en fonction de leurs activités (stationnaire, travail, déplacement). Six zones sont modélisées par des cercles ou des ellipses correspondant à l'état des robots ou des humains.

2.4.4 Navigation

Dans cette section, différentes méthodes de génération de mouvement de navigation sont classées en fonction des tâches pour lesquelles ces méthodes sont conçues. Tout d'abord les techniques pour la navigation libre sont présentées. Ensuite nous décrivons des méthodes pour les cas particuliers de navigation dont les principales sont (1) le passage d'un couloir où une personne et un robot se croisent dans un couloir ; (2) le robot suivant une personne, et (3) le maintien d'une formation.

Navigation Libre

Dans de nombreuses applications, le robot est amené à naviguer dans un environnement en présence d'humains sans forcément avoir à interagir avec l'homme. Dans [Bennewitz 05], Bennewitz et al. ont présenté une stratégie de navigation utilisant des modèles de mouvement humains. Le robot prédit le mouvement des humains et tente d'éviter les chemins prévus. Le but



(a) safety (b) visibility

(c) arm confort

FIGURE 2.6 – Représentation de l'humain sous forme carte de coût dans [Sisbot 07b].

du robot est donc de minimiser le risque d'interférence avec les humains. Si la trajectoire calculée du robot croise l'humain et provoque un blocage, le robot ralentit jusqu'à s'arrêter afin de laisser le passage à l'humain. Contrairement à cette approche, dans [Sasaki 06] les auteurs proposent une méthode où le robot emprunte les mêmes chemins que les humains afin de se comporter de manière plus humaine. Le robot peut alors gêner les humains mais ce comportement est très utile lorsque l'environnement et l'activité posent des contraintes sur la navigation comme dans un théâtre où les personnes doivent éviter de passer devant l'écran.

Dans ces travaux le robot prévoit les mouvements des humains en essayant de ne pas franchir ou suivre les chemins empruntés par les humains mais ne prend pas explicitement en compte le danger ou la gêne qu'il peut occasionner. Il n'est cependant pas nécessaire de prendre en compte l'homme de manière explicite pour produire un comportement sûr. Dans [Madhava Krishna 06], les auteurs ont présenté une méthode proactive de planification de chemin où la sécurité est garantie pendant le mouvement du robot. Le planificateur prend en compte les capacités des capteurs omnidirectionnels à portée limitée du robot. Le planificateur raisonne également sur les vitesses maximales possibles des objets mobiles pouvant être cachés par des obstacles statiques. Le robot s'assure qu'aucune collision ne se produira si un obstacle (humain ou pas) apparaît au moment le plus inattendu. En ne permettant pas au robot de trop s'approcher de zones où le risque de collision est important et de ralentir jusqu'à disparition du danger, la planification produit des déplacements très similaires à ceux effectués par les humains.

Ces approches sont suffisantes pour assurer la sécurité mais ne permettent pas un comportement du robot optimal concernant le confort de l'humain. Une approche qui fut la première technique de planification à prendre en compte l'homme de manière explicite a été développée

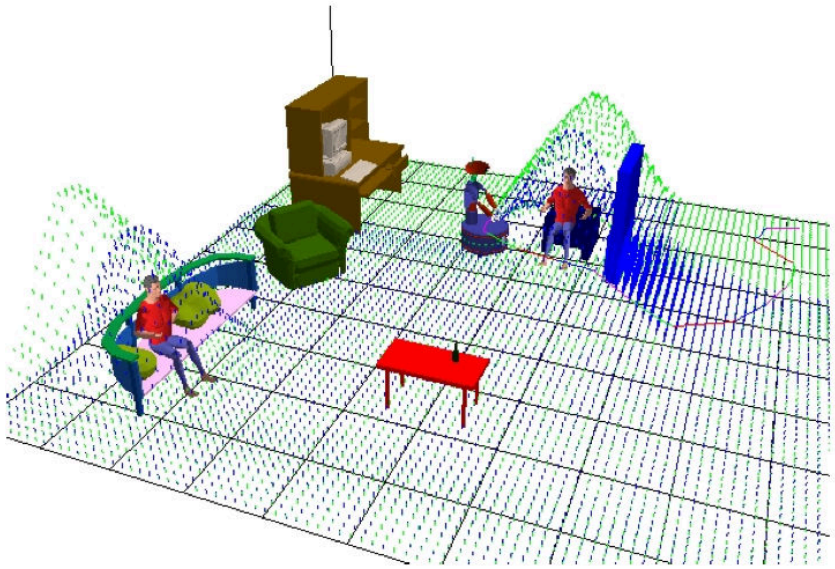


FIGURE 2.7 – Distance, visibilité et zone cachée déterminées sous forme de carte de coût dans [Sisbot 07a]

dans [Sisbot 07a]. Dans ce travail le planificateur raisonne sur un ensemble de cartes de coût qui modélisent les notions "proxemiques". Trois critères principaux sont combinés pour produire un mouvement sûr : la distance, la visibilité et les zones cachées. La Figure 2.7 montre les cartes de coût évaluées pour un espace de travail comprenant deux humains assis sur des fauteuils. Des chemins de bas coût sont ensuite calculés grâce à un algorithme A* comme illustré sur la Figure 2.8. Par la suite, différentes approches ont utilisé des variantes de RRT pour calculer des chemins human-aware sur des cartes de coût similaires afin de planifier avec des contraintes dynamiques ou différentielles. Dans [Svenstrup 10], les auteurs utilisent une variante de RRT dans laquelle un seuil de coût est pris en compte pour rejeter les noeuds avec un coût trop important lors de l'expansion du RRT. Les noeuds trop éloignés dans le temps sont également rejetés pour favoriser l'amélioration des trajectoires dans une fenêtre de temps court. Une autre approche est développée dans [Scandolo 11b], où la connexion au noeud le plus proche est faite avec un biais vers les noeuds de bas coûts.

Dans [Lam 11] un travail complet d'analyse de l'interaction homme-robot et robot-robot définit six règles de coexistence. De ces règles découlent six espaces qui représentent l'homme et le robot dans les situations stationnaires, de travail ou de mouvement. Il en résulte un diagramme d'état permettant de déterminer le comportement idéal du robot. Les auteurs établissent une séparation entre planificateur global et planificateur local permettant un comportement réactif du robot. Plus récemment, l'approche par carte de coût de [Sisbot 07a] a été adaptée au croisement avec une personne dans [Kruse 12]. L'approche initiale par planification sur une carte de coût statique n'est pas adaptée pour prendre en compte des humains en mouvement. Les auteurs séparent également la navigation en deux composants : un planificateur global et un planificateur

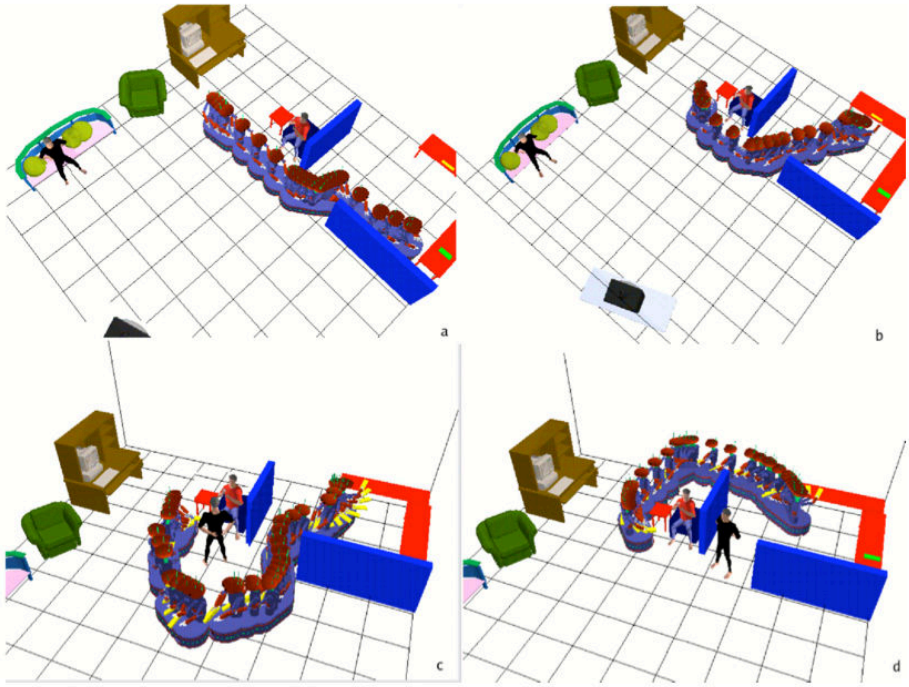


FIGURE 2.8 – Chemin calculé par HAMP (human-aware motion planner) in [Sisbot 07a]

local qui adapte la vitesse du robot de manière réactive. Il en résulte un comportement réactif du robot qui ralentit pour laisser passer l'humain lors d'un croisement. Ce type de comportement est plus lisible que celui généré en considérant des coûts statiques et plus semblable à celui observé chez les humains. Dans [Rios-Martinez 12], les auteurs ont étendu l'approche afin de considérer la tâche et l'interaction entre deux humains. Le principe, mis en évidence dans d'autres travaux récents [Kendon 10], tend à montrer que les humains préfèrent être moins dérangés par un robot qui pénètre leur champs de vision quand ils ne sont pas associés à la tâche ou lorsqu'il pénètre l'espace d'interaction entre deux humains. La planification de mouvement mise en oeuvre dans [Rios-Martinez 12] est basée sur une méthode d'optimisation stochastique qui permet de prendre en compte le mouvement anticipé de l'humain afin de produire des mouvements de navigation confortables.

Croisement dans un couloir

Les couloirs dirigent et restreignent les déplacements et les distances inter-personnelles sont réduites à l'espace intime [Hall 66]. Un robot dans un environnement humain devra très probablement emprunter ces passages étroits pour se déplacer entre les différentes pièces. Etant donné que la distance entre les personnes et les robots se raccourcit dans un couloir, des méthodes traitant explicitement ce type d'environnement sont nécessaires pour assurer la sécurité des humains et également celle du robot.

Les premières méthodes de passage de couloir imitent le comportement humain. Dans [Yoda 97] le robot s'écarte de son chemin quand il s'approche d'une personne. Les distances seuils entre

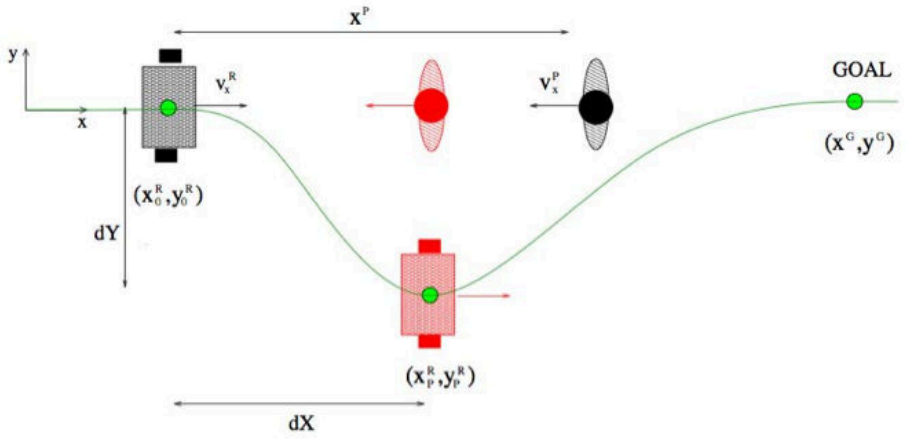


FIGURE 2.9 – Robot passant une personne dans un couloir [Pacchierotti 06a]

l'humain et le robot pour lesquels le robot commence à dévier et pour lesquels il passe à côté de l'homme, sont obtenues à partir d'études utilisateurs. La même méthode est utilisée avec des paramètres différents pour une personne qui marche ou qui effectue une course à pied (à petits pas). Dans [Pacchierotti 06a], Pacchierotti et al. ont étendu cette méthode avec la capacité de s'adapter aux changements de vitesse de la personne, ce qui a permis une interaction dynamique entre le robot et la personne (voir Figure 2.9). Le mouvement du robot est également contrôlé par deux paramètres empruntés à la proxémique : la distance de signalisation, qui représente la distance entre l'humain et le robot lors de l'activation du comportement de croisement et la distance latérale, qui représente le seuil que le robot maintient quand il passe à côté de l'humain.

Suivi d'une personne

Un robot qui suit une personne est un scénario très commun et largement étudié [Yoshimi 06]. Dans [Takemura 07], les auteurs ont introduit un algorithme de type champ de potentiel afin de suivre une personne et d'éviter simultanément les obstacles. Dans cette méthode, la personne se voit attribuer une force attractive et les obstacles des forces répulsives. Ces méthodes simples sont limitées à des cas simples. Par ailleurs, une méthode plus complète a été introduite par Hoeller et al. dans [Hoeller 07] où le robot construit un arbre EST (extensive-space tree) [Hsu 97] autour de sa position courante et un chemin sans collision de coût minimal vers l'humain est extrait. Une contribution intéressante de ce travail est l'introduction de l'estimation du mouvement humain. Le robot estime une cible à court terme pour l'humain, et calcule une trajectoire probable de l'homme en utilisant une méthode de champ de potentiel similaire à celle de [Takemura 07]. Le robot met à jour en permanence l'arbre en fonction de la trajectoire de l'humain trouvée précédemment.

Ces travaux considèrent le robot comme suiveur de l'humain. Dans un scénario de guidage du robot, le robot a l'initiative du mouvement et l'humain suit. Puisque le robot est en face de l'humain et ouvre la voie, l'interaction se situe seulement au niveau de la communication et pas au

niveau du mouvement. Néanmoins dans [Martinez-Garcia 05], Martinez-Garcia et al. présentent un scénario où une équipe de robots guident un groupe d'humains. Les robots surveillent le centre de gravité du groupe humain et contrôlent l'accélération angulaire du centre de gravité du groupe. L'intérêt de ce travail du point de vue de l'interaction homme-robot est de ne pas nécessiter de communication explicite entre les humains et les robots.

Maintien de formation

Dans notre vie quotidienne, il existe de nombreuses situations dans lesquelles nous avons besoin d'établir une formation spécifique et de la maintenir pendant une période de temps. Cette formation peut être une condition sine qua non de notre activité comme dans le cas de patienter dans une file ou de maintenir une formation circulaire afin que chaque personne puisse interagir avec toutes les autres. On peut donc chercher à reproduire le même type de comportement pour un robot qui partage l'espace de travail du groupe et doit s'intégrer à la formation d'une manière naturelle.

Dans [Nakauchi 02], Nakauchi et Simmons présentent un robot qui dans un contexte de file d'attente est capable d'aller à la tête de la ligne et de commencer à se déplacer vers la fin de la ligne face aux humains. Dans ce travail, les humains sont modélisés non seulement par leurs positions et orientations, mais aussi par leurs distances personnelles. La distance personnelle est modélisée comme une ellipse plus large vers l'avant. En tenant compte de ce modèle, le robot tente de maintenir une distance suffisante pour éviter toute perturbation de la personne en face et assez petite pour éviter les personnes en fin de ligne.

Un autre exemple de maintien d'une formation sociale peut être observé dans un groupe en discussion. Le maintien en une formation circulaire permet en général à chaque membre du groupe de voir les autres et de leur parler. Althaus et al. présente dans [Althaus 04] un robot qui se joint à une formation circulaire de manière naturelle et maintient cette formation tout au long de son activité. Avec ces représentations, le robot est capable de maintenir sa formation en tenant compte des nouveaux arrivants et des sortants.

2.4.5 Manipulation

Un robot doté de fonctionnalités de manipulation sera appelé à interagir étroitement avec les humains. Dans ce type d'interaction, le robot remet des objets aux humains, transporte des objets pour les humains et manipule des objets avec les humains. Certaines techniques de planification probabiliste pour chaine fermée [Cortés 05] ont été proposées pour planifier des mouvements de manipulation où le robot et l'humain portent une charge en commun. Cependant l'étude des tâches de manipulation en interaction avec l'homme se concentre essentiellement sur les tâches d'échange d'objet [Edsinger 08]. La première et la seule approche à notre connaissance à développer une approche "human-aware" à la planification de mouvements de manipulation se trouve dans [Sisbot 10, Sisbot 12]. Les auteurs développent une technique basée sur les cartes de coût introduites dans [Sisbot 07b] combinant des techniques de grille à la cinématique inverse généralisée [Nakamura 87].

2.4.6 Transfert d'objet homme-robot

Le transfert d'objet est une capacité essentielle que l'ont peut attendre d'un robot compagnon. Différents aspects ont été étudiés comme le placement relatif homme-robot, le mouvement du bras, sa trajectoire et sa dynamique, la coordination ou le signalement de l'échange mais également la sécurité de l'humain et son confort.

Les premières contributions [Shibata 95] ont porté sur les profils de vitesse du donneur et du receveur en 2d. Les auteurs ont montré que les profils à jerk minimal sont adaptés aux modèles de mouvement de l'humain proposés par Flash and Hogan [Flash 85]. Ils ont également mesuré le temps à partir duquel la planification de mouvement est prise en compte.

L'approche par planification pour le transfert d'objets robot-homme proposé par [Kajikawa 95] repose sur une méthode pour calculer les mouvements par champs de potentiel dans l'espace des configurations du robot. A partir de l'étude des transferts d'objet humain-humain ces travaux ont aussi montré que les gestes ressemblant à l'humain amélioraient la lisibilité de l'intention du robot. La lisibilité dans le transfert d'objet a été évaluée dans des travaux plus récents, dans [Huber 08] les auteurs ont évalué l'importance d'un mouvement imitant celui d'un humain en comparant les transferts d'objet humain-humain avec des transferts d'objet robot-humain, montrant que les profils à jerk minimal nécessitent un temps de réaction inférieur. Un autre aspect important concernant la sécurité de l'interaction a été étudié dans [Kulic 07a], où Kulic et Croft ont détecté des seuils "affectifs d'éveil", utilisant des signaux physiologiques et un moteur d'interférence pour évaluer la qualité des deux planificateurs de mouvement. Les sujets humains de l'expérience ont montré moins d'anxiété et de surprise que pour un planificateur sûr.

Dans [Edsinger 08], Edsinger et Kemp ont étudié des tâches homme-robot coopératives et ont développé un système complet de manipulation interactive. Ils ont également montré que les humains ont tendance à s'adapter à la forme du robot et à ses capacités limitées de préhension en sélectionnant une position et orientation de l'objet qui facilite le transfert d'objet.

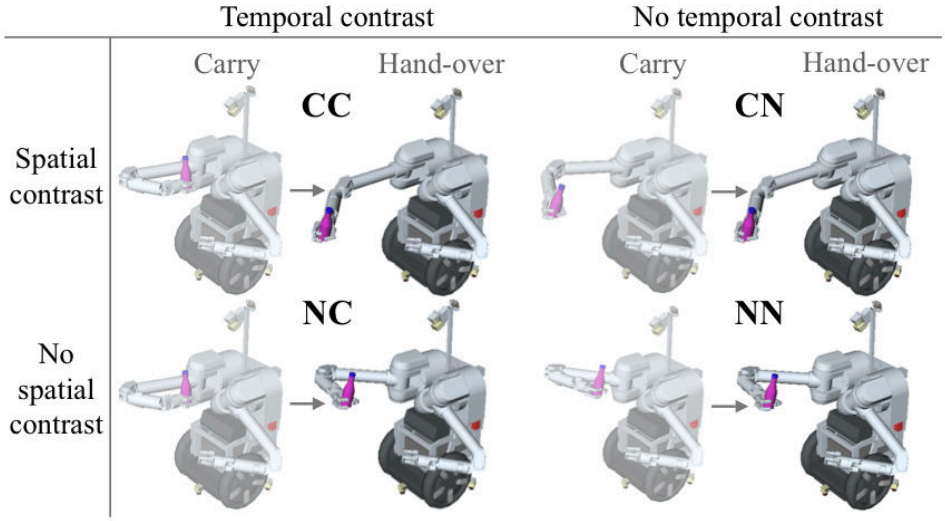


FIGURE 2.10 – Contraste temporel et spatial des postures de transfert d'objet dans [Cakmak 11]

Plus récemment Cakmak et al [Cakmak 11] ont introduit la notion de contraste pour générer les postures d'échanges d'objet. En effet les postures du robot véhiculent généralement mal l'intention de donner l'objet car les postures pour porter l'objet et les postures d'échanges sont souvent trop semblables. La notion de contraste a été dérivée en contraste spatial et contraste temporel. Le contraste spatial est défini par la distinction entre la situation (position et orientation) avec laquelle l'objet est présenté à l'humain comparée à celles prises pour d'autres actions que le robot pourrait faire avec l'objet en préhension. Le contraste temporel traduit la différence entre la situation de l'objet et celle qui la précède. La Figure 2.10 présente quatre transitions avec et sans contraste temporel et spatial. Les postures qui comportent un contraste spatial et temporel (en haut à gauche) traduisent mieux l'intention de donner l'objet que celles qui ne comportent aucun contraste (en bas à droite).

Finalement, dans [Sisbot 10] une approche utilisant la combinaison de différents critères a été développée pour trouver la position de transfert d'objet. L'idée est de chercher le minimum dans une grille qui traduit les critères liés à la sécurité, la visibilité et le confort du bras. Cette approche a été couplée à un générateur de mouvement souple qui limite le jerk le long de la trajectoire et a été évaluée dans [Dehais 11] avec une combinaison de métriques subjectives et objectives. La conductance galvanique de la peau, l'activité du deltoïde et l'activité oculaire ont été mesurées et les réponses à un ensemble de questions ont été étudiées.

3

Planification de mouvement en interaction avec l'homme

Dans des travaux antérieurs, [Sisbot 07a, Sisbot 07b, Sisbot 12] Sisbot et al. ont présenté un planificateur de mouvement qui prend explicitement en compte des contraintes d'interaction homme-robot telles que leur distance relative, le champ de vision de l'humain et sa posture, pour synthétiser des mouvements de navigation et de manipulation. Ce planificateur était basé sur des études de cas d'interaction homme-robot [Koay 07], ainsi que sur les théories existantes sur le partage de l'espace inter-humains [Hall 63]. La méthode proposée a été à notre connaissance la première initiative à aborder le problème du partage de l'espace intelligent entre l'homme et le robot par la «planification». Les contraintes HRI étaient représentées par des fonctions de coût basées respectivement sur le modèle cinématique de l'homme, son champ de vision et ses capacités de manipulation. Cette représentation reposait sur des cartes de coûts définies dans l'espace de travail. La planification de mouvement était effectuée en utilisant des techniques de grilles, et la cinématique inverse généralisée pour faire suivre le chemin de l'objet par le robot.

Bien que cette approche découplée soit suffisante en l'absence de fortes contraintes dans l'espace de travail, elle peut échouer dans un environnement encombré comme celui illustré par la Figure 3.1. De plus l'humain étant par nature mobile, il est important de considérer des méthodes de planification de mouvement qui permettent de prendre en compte la dynamique introduite dans l'espace de travail.

Dans ce chapitre, nous étendons les capacités du planificateur en utilisant des algorithmes de planification basés sur l'échantillonnage aléatoire. Cette approche permet une planification dans l'espace des configurations du robot pour trouver des mouvements prenant en compte l'homme dans des environnements contraints. En effet, les méthodes de planification de mouvement par échantillonnage [Choset 05b, LaValle 06] sont capables de traiter des problèmes complexes dans

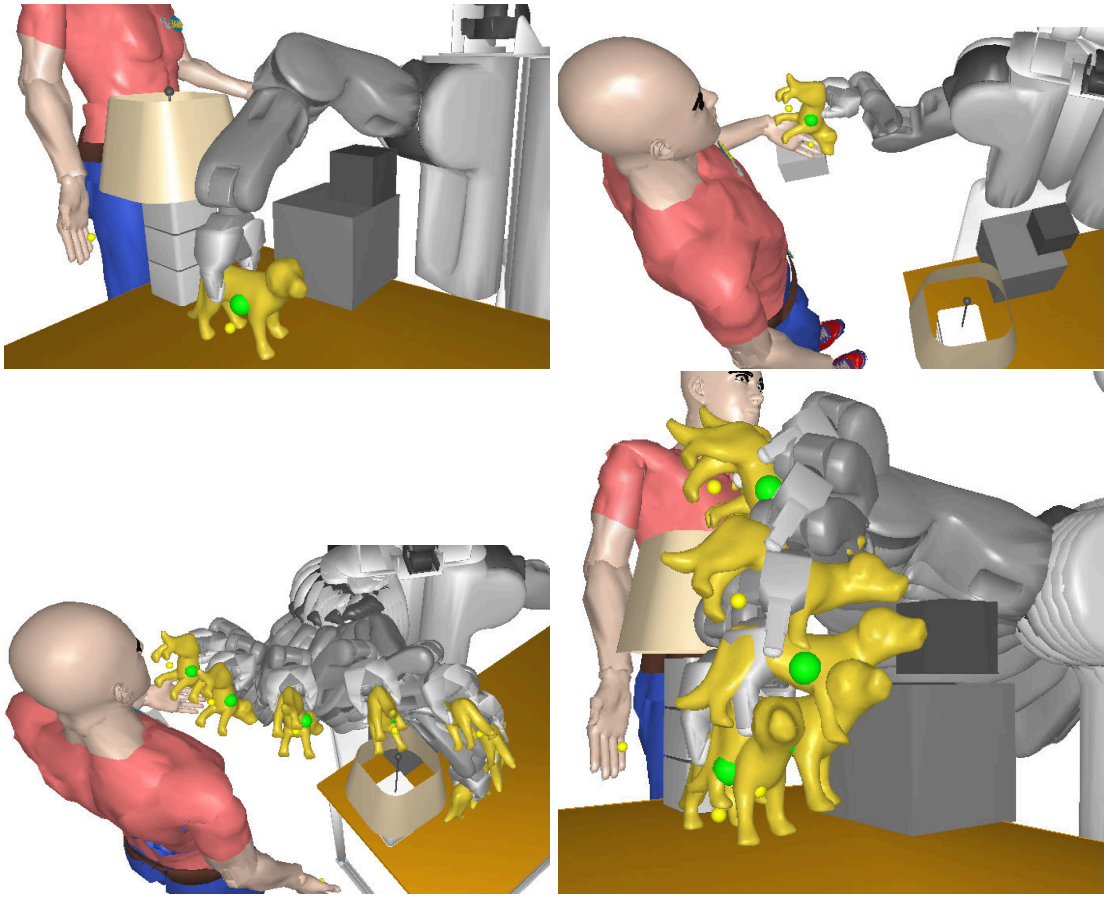


FIGURE 3.1 – Un espace de travail encombré présente un problème de planification de mouvement difficile pour le robot et la présence de l'humain ajoute des contraintes supplémentaires.

des espaces de haute dimension. Toutefois, elles fonctionnent dans un espace des configurations généralement binaire, visant à trouver des solutions faisables sans collision plutôt que des chemins optimaux. En outre, en raison de leur nature probabiliste, les chemins solutions sont généralement de faible qualité. Une phase de post-traitement est couramment utilisée pour les améliorer localement vis à vis de différents critères tels que la longueur et la distance aux obstacles.

La méthode proposée dans ce chapitre s'appuie sur deux algorithmes récents. T-RRT [Jaillet 10], qui calcule des chemins de bonne qualité considérant une fonction de coût définie sur l'espace des configurations, et STOMP [Kalakrishnan 11] qui déforme localement une trajectoire en considérant des contraintes internes et externes liées à la loi de commande et aux obstacles. Les solutions fournies par T-RRT sont donc améliorées par STOMP qui permet de par son fonctionement "anytime" de prendre en compte de petits changements dans l'espace de travail.

Après avoir introduit ces deux méthodes, nous présentons une description détaillée des contraintes HRI, en particulier la contrainte de "confort du bras" utilisée dans [Sisbot 07b] pour le calcul du point de transfert d'objet. Cette contrainte est considérée ici pendant la planification de mouvement afin de produire des trajectoires qui facilitent la prise de l'objet à tout moment. Nous discutons ensuite des adaptations nécessaires de T-RRT et STOMP pour leur application aux

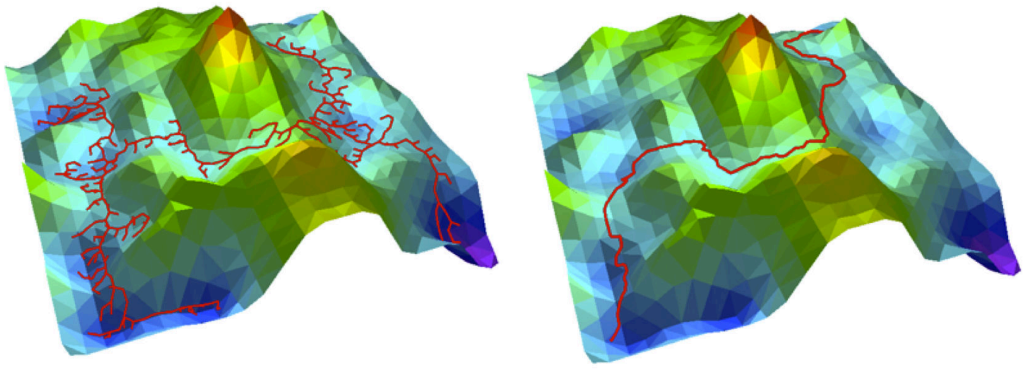


FIGURE 3.2 – T-RRT développé sur une carte de coût 2D (à gauche). Le test de transition favorise l'exploration des régions de bas coût, résultant en un chemin de bonne qualité (à droite).

fonctions de coût HRI. Nous étudions tout d'abord T-RRT, nous introduisons différents espaces de recherche (articulaire ou cartésien) et une version bidirectionnelle de l'algorithme. Puis dans une deuxième partie, nous étudions l'adaptation de la méthode classique de lissage "shortcut" [Berchtold 94] et une méthode d'optimisation par perturbations aléatoires, que nous avons proposé et qui permet de s'affranchir de certaines limitations de "shortcut". Enfin, nous présentons les adaptations nécessaires pour combiner T-RRT et STOMP. Ces différentes techniques sont analysées par des expériences en simulation en fin de chapitre.

3.1 Techniques de planification de mouvement

Nous considérons deux techniques récentes (T-RRT et STOMP), brièvement introduites dans l'état de l'art, qui permettent de traiter des problèmes de planification de mouvement hautement dimensionnés. Ces deux approches prennent en compte des fonctions de coût générales définies dans l'espace des configurations du robot et ne nécessitent pas de connaître leur gradient. Elles opèrent cependant de manière distincte.

T-RRT est une méthode globale qui explore l'espace des configurations en suivant les vallées de la carte de coût. STOMP est une méthode locale qui déforme itérativement un chemin solution par estimation stochastique du gradient dans l'espace des trajectoires. Cette section présente ces deux méthodes plus en détail.

3.1.1 T-RRT

L'algorithme T-RRT [Jaillet 10] est un algorithme de planification permettant d'obtenir des chemins de bonne qualité dans des espaces de coût de haute dimension. Il ne s'agit pas réellement de planification optimale mais T-RRT possède de bonnes propriétés et produit rapidement des solutions de faible coût. Pour cela, il tire parti des performances de deux méthodes. Tout d'abord, il bénéficie de la force d'exploration des planificateurs de type RRT résultant de leur biais d'expansion vers les grandes régions de Voronoï de l'espace exploré. Il intègre également certaines fonctionnalités des méthodes d'optimisation stochastique [Robert 99], qui consistent à

appliquer des tests de transition pour accepter ou rejeter des états potentiels. Ces transitions forcent la recherche à suivre les vallées et les cols de la carte de coût afin de trouver des chemins solutions de faible coût (voir Figure 3.2). Plusieurs critères peuvent être utilisés pour mesurer la qualité d'un chemin à partir d'une fonction de coût paramétrique : e.g. le coût maximal, le coût moyen, l'intégrale des coûts le long du chemin, ou le travail mécanique. L'algorithme T-RRT vise à trouver des chemins qui minimisent ce critère du travail mécanique. De plus, il satisfait simultanément d'autres mesures de qualité telles que l'intégrale des coûts [Jaillet 10].

Algorithm 3.1: Transition-based RRT

```
begin
    T ← InitialiseArbre(q_init);
    while not ConditionArret(T, q_goal) do
        q_rand ← EchantilloneConf(CS) ;
        q_near ← MeilleurVoisin(q_rand, T);
        q_new ← ExtentionArbre(T, q_rand, q_near);
        if q_new ≠ NULL
        and TestDeTransition(c(q_near), c(q_new), d_near−new)
        and ControlExpansionMini(T, q_near, q_rand) then
            AjouNouvNoeud(T, q_new);
            AjouNouvArrête(T, q_near, q_new);
end
```

L'algorithme 3.1 montre le pseudo-code du planificateur T-RRT. De même que pour la version "Extend" de l'algorithme RRT, une configuration q_{rand} est choisie au hasard. Elle donne à la fois le nœud q_{near} le plus proche dans l'arbre qui est étendu, et la direction d'extension. L'extension de q_{near} est effectuée vers q_{rand} avec un pas d'extension δ. Celui-ci doit être assez petit pour éviter de manquer des pics dans la fonction de coût. Ainsi, si la nouvelle portion du chemin conduit à une collision, aucune configuration n'est retournée et l'extension échoue indépendamment des coûts associés. Ce processus d'extension assure le biais vers des régions inexplorées de l'espace. L'objectif de la deuxième étape est de filtrer les configurations q_{new} en fonction de leur coût avant leur insertion dans l'arbre. Ceci permet de trouver des chemins de faible coût.

Le filtrage est effectué par la fonction `TransitionTest`, détaillée dans l'algorithme 3.2. La probabilité d'acceptation d'une nouvelle configuration est définie en comparant son coût à celui du noeud parent dans l'arbre. Le test s'appuie sur le critère de Metropolis, couramment utilisé par les méthodes d'optimisation stochastique, avec une probabilité de transition p_{ij} qui sanctionne les mouvements dont le coût augmente. Elle est définie comme suit :

$$p_{ij} = \begin{cases} exp(-\frac{\Delta c^*_{ij}}{K \cdot T}) & \text{if } \Delta c^*_{ij} > 0 \\ 1 & \text{otherwise.} \end{cases} \tag{3.1}$$

Algorithm 3.2: Test de transition

```
begin
    if c_j < c_i then
        return true ;
    if Random(0,1) < MetropolisTest(c_i,c_j,T) then
        T ← T/α ;
        nFail ← 0 ;
        return true ;
    else
        if nFail > nFail_max then
            T ← T * α ;
            nFail ← 0;
        else
            nFail ← nFail + 1;
    return false ;
end
```

où $\Delta c^*_{ij} = (c_j - c_i)/d_{ij}$ est la pente sur la carte de coût entre q_{near} et q_{new}, T est un paramètre de température et K un facteur constant. Le calcul de p_{ij} est effectué dans la fonction `MetropolisTest`. L'ajout de la configuration dans le graphe dépend de p_{ij} uniquement dans le cas d'une pente positive. Dans ce cas, si le test de Metropolis échoue et qu'un nombre $nFail_{max}$ d'échecs du test consécutif est atteint, la température T est élevée d'un facteur constant $\alpha = 2$. Elle est diminuée quand le test réussit. Ce principe permet un réglage automatique qui permet de contrôler la puissance de filtrage, conduisant l'exploration à travers les passages par les cols les moins élevés reliant des régions de faible coût.

Enfin, la fonction `MinExpandControl` force le planificateur à maintenir un taux minimal d'expansion vers les régions inconnues de l'espace, évitant ainsi la possibilité de blocage de la recherche.

Lissage

Une étape de lissage est communément employée sur les chemins issus des planificateurs par échantillonnage aléatoire. La méthode "shortcut" [Berchtold 94] dont le pseudo code est présenté sur l'algorithme 3.3, lisse le chemin de manière itérative. Elle consiste à sélectionner aléatoirement des paires de configurations sur le chemin $\mathcal{P}$ puis à les connecter par un segment de droite. La partie entre ces deux configurations est remplacée par un nouveau segment plus court si il est valide, (i.e. sans collisions).

Cette technique surpasse l'heuristique d'élagage de chemin qui considère non seulement les nœuds sur le chemin, mais aussi toutes les configurations intermédiaires. Les configurations peuvent être choisies de manière aléatoire [Kavraki 98, Chen 98] ou déterministe [Baginski 97]. Dans la section 3.4 nous présentons des adaptations de cette méthode pour les problèmes de planifi-

cation traités par T-RRT ce qui nécessite la prise en compte du coût.

Algorithm 3.3: Shortcut

```
begin
    while not ConditionArret() do
        (q1, q2) ← P.retourneDeuxConfig();
        LP ←segment(q1, q2) ;
        if estValide(LP, q1, q2) then
            P.remplaceLaPortion(LP, q1, q2) ;
end
```

3.1.2 STOMP

STOMP (Stochastic Trajectory Optimization for Motion Planning) est un algorithme qui optimise itérativement une trajectoire discrétisée. Le gradient dans l'espace des trajectoires est estimé de manière stochastique. Il est réparti sur l'ensemble des points intermédiaires afin d'obtenir des mises à jour lisses.

A chaque itération, un ensemble de trajectoires bruitées est généré. Ces trajectoires sont ensuite évaluées pour déterminer leur coût qui est utilisé pour mettre à jour la solution courante. Il n'est donc pas nécessaire de connaitre le gradient de la fonction de coût, ce qui est particulièrement utile pour traiter des fonctions de coût non-différentiables ou non-lisses.

STOMP considère le problème d'optimisation suivant :

$$\min_{\tilde{\xi}} \mathbb{E}\left[f_{prior}(\tilde{\xi}) + f_{obst}(\tilde{\xi}) \right]$$

où $\tilde{\xi} = \mathcal{N}(\xi, \Sigma)$ est un vecteur de paramètres qui spécifie N configurations intermédiaires du robot le long de la trajectoire, avec une variance Σ et une moyenne ξ. Le problème d'optimisation consiste donc à minimiser l'espérance d'une fonction de coût combinant des "forces" internes et externes. Ces forces agissent sur la trajectoire pour maintenir sa cohésion et garantir une certaine distance aux obstacles. Cette formulation du problème a été introduite dans [Quinlan 93] pour résoudre des problèmes de planification de mouvement de navigation par déformation de trajectoire.

La fonction f_{prior} est définie de la manière suivante :

$$f_{prior}(\xi) = \frac{1}{2} \xi^T R \xi$$

avec $R = K_2^T K_2$. K_2 étant la matrice aux différences finies suivante :

$$K_2 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ -2 & 1 & 0 & \dots & 0 & 0 & 0 \\ 1 & -2 & 1 & \dots & 0 & 0 & 0 \\ & \vdots & & \ddots & 0 & & 0 \\ 0 & 0 & 0 & \dots & 1 & -2 & 1 \\ 0 & 0 & 0 & \dots & 0 & 1 & -2 \\ 0 & 0 & 0 & \dots & 0 & 0 & -1 \end{bmatrix}$$

La multiplication par K_2 du vecteur de paramètre estime l'accélération à chaque configuration intermédiaire par la formule $x_{n-1} - 2 * x_n + x_{n+1}$. f_{prior} est donc la somme des accélérations au carré le long de la trajectoire discrétisée.

La deuxième partie de la fonction objectif, f_{obst} est la somme des coûts de pénétration dans les obstacles exprimée de la manière suivante :

$$f_{obst} = \sum_{i=0}^{N} cost(\xi_i)$$

où N est le nombre de configurations dans le vecteur de paramètres ($\xi_i = q$) et *cost* est la fonction de coût de pénétration dans les obstacles.

Le problème d'optimisation peut donc s'exprimer comme la minimisation de la somme des accélérations le long de la trajectoire additionnée à la somme d'un coût de pénétration dans les obstacles de chaque configuration intermédiaire, de la manière suivante :

$$\min_{\tilde{\xi}} \mathbb{E}\left[\frac{1}{2}\xi^T R \xi + \sum_{i=0}^{N} cost(\xi_i)\right]$$

Gradient stochastique

En posant que le gradient de la fonction objectif doit être nul au minimum :

$$\nabla_{\tilde{\xi}}\left(\mathbb{E}\left[f_{prior}(\tilde{\xi}) + f_{obst}(\tilde{\xi})\right]\right) = 0$$

On peut montrer que la règle de mise à jour se résume à :

$$\mathbb{E}(\tilde{\xi}) = -R^{-1}\delta\hat{\xi_G}$$

avec $\hat{\xi_G}$ étant l'espérance du gradient de la fonction de coût :

$$\hat{\xi_G} = \mathbb{E}\left(\nabla_{\tilde{\xi}}\left[f_{obst}(\tilde{\xi})\right]\right)$$

Des approches précédentes [Ratliff 09] ont utilisé le gradient analytique pour dériver une règle de mise à jour itérative. Ici, l'estimation du gradient s'inspire de travaux antérieurs [Dayan 97]

ainsi que de travaux récents dans le domaine de l'apprentissage par renforcement d'intégrale de chemin [Theodorou 10]. La mise à jour peut s'exprimer de la manière suivante :

$$\hat{\xi_G} = \int exp\Big(-\frac{1}{\lambda} S(\xi) \Big) \delta\xi d(\delta\xi)$$

avec $S(\xi) = f_{obst}(\xi) = \sum_{i=0}^{N} cost(\xi_i)$. La procédure d'optimisation est statique, c'est à dire qu'elle ne nécessite pas l'exécution de la trajectoire contrairement à l'approche d'apprentissage par renforcement. Cependant l'estimation du gradient stochastique est fortement connectée à la façon dont la valeur de la fonction est calculée pour l'intégrale de chemin dans le formalisme de contrôle optimal stochastique [Astrom 70, Bertsekas 78].

Dans ces travaux, les commandes sont calculées pour chaque état x_{ti} comme étant $\delta\hat{u} = \int p(x)\delta u$ où δu sont les commandes échantillonnées et $p(x)$ correspond à la probabilité de chaque chemin τ_i commençant à x_{ti} et terminant à l'état final x_{tN}. Cette probabilité est définie comme $p(x) = exp(-\frac{1}{\lambda}S(\xi))$ avec $S(\tau_i)$ étant le coût du chemin τ_i. Les chemins avec un coût élevé ont donc une contribution moins importante aux commandes optimales que les chemins avec un coût faible.

Algorithm 3.4: STOMP

input : The path parameters ξ ;
output : The path parameters ξ ;
begin
  $R^{-1} \leftarrow (A^T A)^{-1}$;
  $M \leftarrow$ `dimensioneChaqueColone`(R^{-1}) ;
  while not `ConditionDArret()` **do**
    `CréeKTrajectoiresBruitées`(R^{-1}) ;
    `CalculCoutDesKTrajectoires`() ;
    for *k = 1 ... K* **do**
      for *i = 1 ... (N-1)* **do**
        $P(\tilde{\xi}_{k,i}) \leftarrow \frac{e^{\frac{-1}{\lambda} cost(\tilde{\xi}_{k,i})}}{\sum_{i=1}^{K}[e^{-\frac{1}{\lambda} cost(\tilde{\xi}_{k,i})}]}$;
    for *i = 1 ... (N-1)* **do**
      $[\delta\tilde{\xi}]_i \leftarrow \sum_{k=1}^{K} P(\tilde{\xi}_{k,i})[\epsilon_k]_i$;
    $\delta\xi \leftarrow M\delta\tilde{\xi}$;
    $\xi \leftarrow \xi + \delta\xi$;
    $c(\xi) \leftarrow f_{prior}(\xi) + f_{obst}(\xi)$
end

Algorithme STOMP

Le pseudo code de l'algorithme STOMP est présenté en 3.4. A chaque itération la fonction `CréeKTrajectoiresBruitées` génère K trajectoires bruitées avec une moyenne de zéro et

une variance R^{-1} dans la fonction (voir Figure 3.3). Ces trajectoires ont un coût associé à la "smoothness" relativement bas et n'induisent pas de divergences aux points initiaux et finaux. Ces trajectoires multidimensionnelles peuvent être multipliées par un paramètre de réglage afin de contrôler l'amplitude du bruit. Ce paramètre est analysé dans la section 4.3. Ensuite, le coût pour chaque configuration $\xi_{k,i}$ du vecteur de paramètre est calculé pour les K trajectoires dans la fonction `CalculCoutDesKTrajectoires`. Les probabilités associées à chaque configuration $P(\tilde{\xi}_{k,i})$ des K trajectoires sont calculées avec la formule suivante :

$$P(\tilde{\xi}_{k,i}) = \frac{e^{\frac{-1}{\lambda} cost(\tilde{\xi}_{k,i})}}{\sum_{i=1}^{K}[e^{-\frac{1}{\lambda} cost(\tilde{\xi}_{k,i}}]}$$

A cette étape un certain nombre de trajectoires bruitées peuvent être conservées et réutilisées pour générer la mise à jour. Dans l'implémentation que nous avons utilisée, K=5 et les 5 meilleures trajectoires sont également réutilisées à l'itération suivante.

Ensuite, pour chaque pas de temps, la mise à jour bruitée $[\delta\tilde{\xi}]_i$ est calculée comme la combinaison convexe de chaque trajectoire bruitée pondérée par la probabilité $P(\tilde{\xi}_{k,i})$. Puis cet incrément $\delta\tilde{\xi}$ est multiplié par une matrice M afin de lisser la mise à jour. La matrice M est formée par mise à l'échelle de chaque colonne de la matrice R^{-1} afin que l'élément le plus grand de chaque colonne ne dépasse pas $1/N$. La nouvelle trajectoire est essentiellement une combinaison convexe des trajectoires bruitées déjà évaluées, c'est à dire qu'il n'y a pas de sauts inattendus vers les parties inexplorées de l'espace d'état en raison d'une évaluation bruitée du gradient. Ces mises à jour de la trajectoire peuvent être considérées comme plus sûres que les mises à jour de descente de gradient standard. La Figure 3.4 montre les trajectoires bruitées générées sur une carte de coût et la trajectoire obtenue après 50 itérations de STOMP.

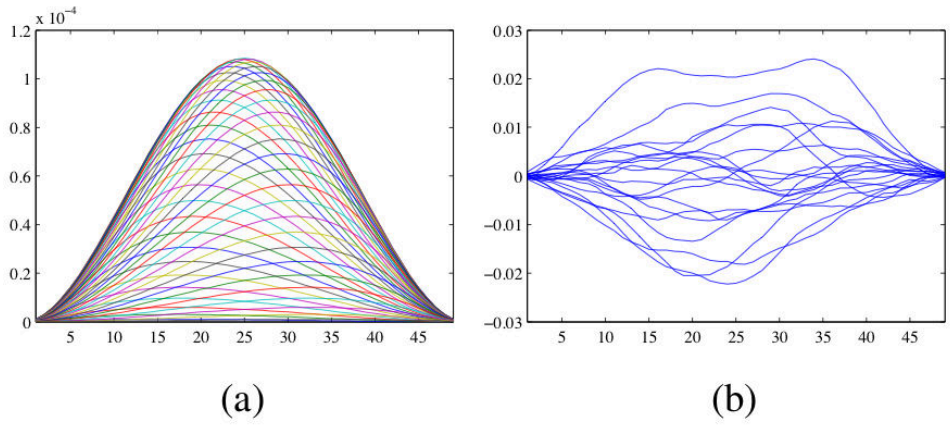
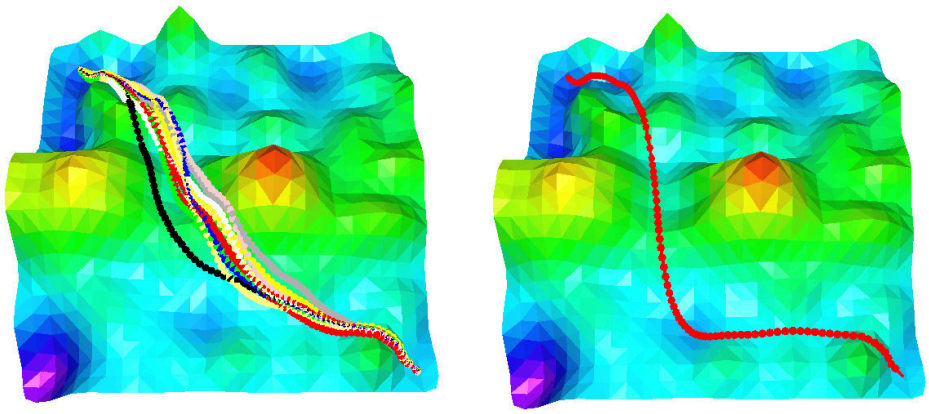


FIGURE 3.3 – (a) Chaque courbe montre une colonne/ligne de la matrice R^{-1}, (b) la génération de 20 trajectoires bruitées monodimenstionelles avec une covariance R^{-1}



(a) Trajectoires bruitées (b) Trajectoire après 100 iterations

FIGURE 3.4 – (a) Les trajectoires bruitées constituées de 100 points de contrôle permettent d'explorer la carte de coût localement. (b) La trajectoire initiale (ligne droite) est déformée afin de mieux prendre en compte les vallées de la carte de coût.

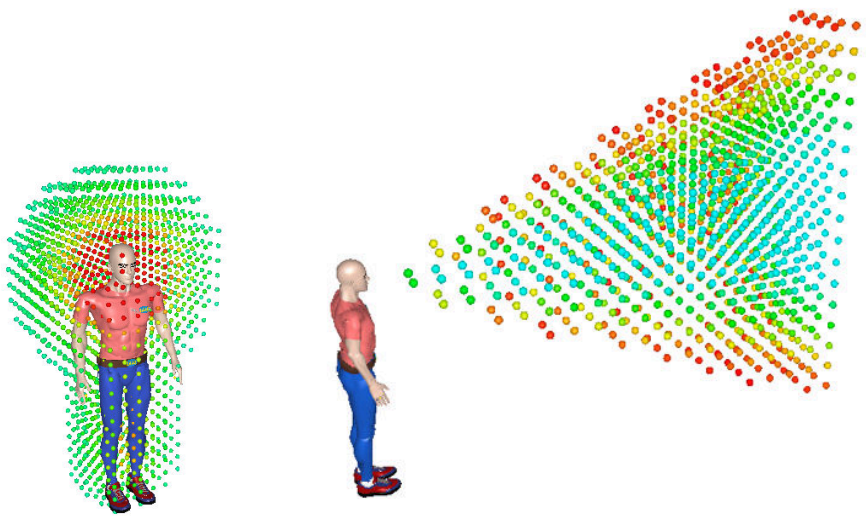
3.2 Contraintes d'interaction Homme-Robot

La présence d'humains dans l'espace de travail du robot génère de nouvelles contraintes pour la planification de mouvements de navigation et de manipulation. Plusieurs contraintes importantes prises en compte dans des travaux antérieurs telles que la sécurité, la visibilité ou le de confort du bras sont détaillées dans cette section.

La première contrainte présentée sur la Figure 3.5(a), appelée contrainte de distance, a pour but principal d'assurer la sécurité de l'interaction en contrôlant la distance entre le robot et l'humain. Seul un volume approximatif délimitant le corps de l'homme sans tenir compte de la géométrie des bras est utilisé pour le calcul de distance. Cette contrainte de sécurité maintient le robot loin de la tête et du corps, l'accent étant mis sur la prévention de tout risque de collision dangereuse entre l'humain et le robot. La théorie proxémique [Hall 63] montre que le dépassement d'un seuil délimitant l'espace intime génère une sensation de peur, voir un sentiment d'insécurité. Par conséquent, un point qui sera proche de l'homme aura un coût de sécurité élevé alors que le coût diminue pour les points plus éloignés jusqu'à un seuil maximal au delà duquel il devient nul.

La deuxième contrainte, appelée contrainte de visibilité, a pour but de limiter l'effet de surprise que peut ressentir l'homme pendant que le robot bouge dans l'espace de travail. Un homme se sent moins surpris si le robot reste visible ce qui résultera en une interaction plus sûre et plus confortable comme cela est montré dans [Sisbot 07a]. Ainsi, chaque point de l'espace de travail a un coût proportionnel à l'angle entre le regard de l'homme et sa position dans l'espace cartésien comme l'illustre la Figure 3.5(b).

La troisième contrainte, appelée contrainte de "confort du bras", a été introduite dans [Sisbot 07b] pour calculer un point de transfert lors de tâches d'échange d'objet entre le robot et l'homme. Cette section présente une description détaillée de cette contrainte, qui est également



(a) Carte de coût de distance (b) Carte de coût de visibilité

FIGURE 3.5 – Contraintes de distance et de visibilité pour lesquelles un coût HRI est attribué à chaque point de l'espace cartésien. Le coût de sécurité est inversement proportionnel à la distance à l'homme alors que le coût de visibilité est fonction de la direction du regard.

considérée par le planificateur de mouvement pour générer des chemins qui facilitent l'échange de l'objet tenu par le robot à tout moment. Pour cette contrainte, le robot doit raisonner sur les capacités d'accessibilité de l'homme et sa cinématique. Le volume supposé atteignable de l'homme peut être pré-calculé en utilisant la cinématique inverse généralisée (GIK). Pour chaque point à l'intérieur du volume atteignable de l'homme, la configuration du torse déterminée par la cinématique inverse est calculée à partir d'une posture de repos donnée. La détection de collision avec l'environnement est ensuite utilisée pour valider ces postures. A chaque posture valide est attribuée un coût de confort grâce au modèle de prédiction de posture humaine introduit dans [Marler 05]. Le confort est estimé par la somme des trois fonctions suivantes :

– La première fonction calcule une distance articulaire à la posture de repos q_{repos} où q est la configuration courante de la personne :

$$f_1 = \sum_{i=1}^{DOF} w_i (q_i - q_i^{repos})^2$$

– La seconde considère l'énergie potentielle du bras qui est définie par la différence des hauteurs des bras et avant-bras avec celles d'une posture de repos (Δz_i) pondérée par une estimation des masses des bras et avant-bras m_i :

$$f_2 = (m_{bras} * g)^2 * (\Delta z_{bras})^2 + (m_{avant-bras} * g)^2 * (\Delta z_{avant-bras})^2$$

– La troisième pénalise les configurations proches des limites articulaires. Pour chaque articulation correspondent des limites articulaires, la limite la plus proche (Δq_i) est prise en

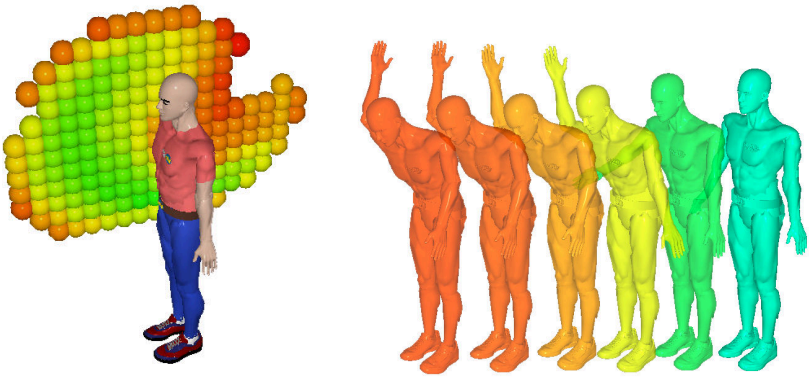


FIGURE 3.6 – Contrainte de confort du bras. A gauche, la carte de coût montre une tranche de la grille des cellules atteignables, le coût des cellules est représenté par des couleurs allant de vert confortable, à rouge pénible pour le bras droit. A droite, des postures évaluées par la fonction coût muskuloskelétique sont représentés en transparence.

compte dans la fonction de coût comme suit :

$$f_3 = \sum_{i=1}^{DOF} \gamma_i \Delta q_i^2$$

Chaque contrainte est représentée par une carte de coût tridimensionnelle et les cartes de coût élémentaires sont combinées par une somme pondérée :

$$c(h,x) = \sum_{i=1}^{N} w_i c_i(h,x),$$

où h est la posture du modèle de l'homme et x le point de l'espace de travail pour lequel le coût est calculé. La carte de coût résultante est illustrée sur la Figure 3.6. Dans l'implémentation actuelle, les poids sont définis empiriquement et les fonctions de coût sont évaluées "à la volée" lors de la planification.

Les planificateurs de [Sisbot 07a, Sisbot 07b] étaient basés sur une recherche directe sur les grilles cartésiennes afin de produire des mouvements agréables de l'effecteur du robot. Ces techniques utilisent des méthodes standards de recherche dans un graphe. Dans [Sisbot 07a], les tâches de navigation étaient effectuées sur la base de grilles 2D explorées par un algorithme A*. L'extension à des tâches de manipulation [Sisbot 07b, Sisbot 12] a conduit à des grilles cartésiennes utilisées pour calculer le chemin de l'effecteur en supposant que le chemin calculé était faisable pour le robot.

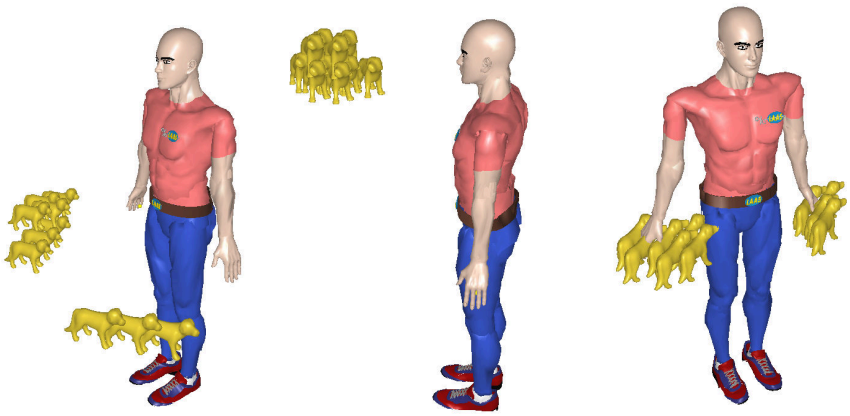
Bien qu'une telle approche découplée soit suffisante en l'absence de fortes contraintes dans l'espace de travail, elle peut échouer dans les situations où la trajectoire prévue pour l'objet ne peut pas être suivie par le robot (cf exemple 3.1). Par conséquent, l'extension à des cartes de coûts définies dans l'espace des configurations du robot est souhaitable. La section 3.4 présente la mise en oeuvre de ces cartes de coût de haute dimension et l'adaptation de techniques de

planification par échantillonnage pour résoudre ce type de problème comprenant des contraintes d'interaction homme-robot.

3.3 Positionnement de l'objet pour transfert

Le calcul de la position de transfert d'objet est essentiel dans les tâches de manipulation interactive. Plusieurs stratégies peuvent être employées. Dans le cas où l'humain est pro-actif il peut lui même signifier un point de transfert avec sa main. Le point de transfert peut être extrait de la position de la main en calculant une position décalée de la position de la main. Après détermination du point, ce dernier est validé par une méthode de cinématique inverse.

Le robot peut aussi calculer une position de manière pro-active en s'appuyant sur les contraintes HRI. Dans [Sisbot 07b, Sisbot 10] les auteurs proposent de combiner les contraintes développées ci-dessus pour trouver un point de transfert qui satisfasse au mieux l'ensemble des contraintes. Il s'agit de trouver la cellule dans une grille de l'espace accessible par l'humain qui minimise la somme pondérée des coûts associés à la distance, à la visibilité et au confort du bras. Lorsque la cellule est trouvée, la configuration du robot est calculée par cinématique inverse. Si aucune configuration n'est trouvée pour une position donnée la cellule candidate est supprimée et on continue avec la cellule suivante. La Figure 3.7, présente différentes positions de transfert calculées en ajoutant plus de poids sur une des contraintes.



(a) (b) (c)

FIGURE 3.7 – 10 meilleures positions calculées pour un transfert d'objet dans l'espace accessible de l'humain. Les positions optimisent (a) la sécurité (b), la visibilité et (c) le confort des bras.

Dans le cas ou le bras du robot comprend 7 DoF, plusieurs stratégies existent pour trouver une cinématique inverse valide. Une méthode analytique peut être utilisée sur 6 DoF qui comprend 8 classes de solutions (en fixant la classe ou en la choisissant aléatoirement). Le septième DoF est ensuite échantillonné de manière aléatoire. On peut également utiliser la cinématique inverse généralisée (GIK) qui est plus couteuse en temps de calcul mais permet de spécifier différentes contraintes comme l'orientation de la tête. Une autre solution consiste a présélectionner

un ensemble de configurations de transfert. Ces configurations peuvent comprendre l'homme et le robot et être triées de manière préalable selon les critères d'interaction. L'ensemble des configurations est ainsi parcouru dans un ordre décroissant de confort pour trouver une configuration sans collision. Cette méthode est utilisé dans le chapitre 4 pour trouver des configurations dans l'espace de travail impliquant les degrés de navigation. Dans le cas d'un échange d'objet comprenant simplement une tâche de manipulation il est préférable d'utiliser la cinématique inverse analytique qui permet de trouver des solutions dans des environnement contraints.

3.4 Adaptations pour la planification en interaction avec l'homme

Dans les travaux de [Sisbot 07a, Sisbot 12] les contraintes HRI ont été prises en compte pour générer des chemins de navigation et de manipulation par des techniques de grille. Ici nous considérons une fonction de coût définie sur l'espace des configurations du robot. Le coût lié à l'interaction homme-robot est défini de manière suivante :

$$c(h,q) = \sum_{i=1}^{N} w_i c_i(h, FK(q)),$$

où h est la configuration de l'homme, q est la configuration du robot et FK la fonction correspondant au modèle cinématique direct du robot. Compte tenu de la carte de coût sur l'espace des configurations résultante, nous adoptons les algorithmes présentés dans la section 3.1 pour trouver des chemins de bonne qualité dans des environnements contraints.

Cette section présente les adaptations nécessaires pour planifier des mouvements de manipulation interactive en considérant cette carte de coût. Dans un premier temps l'approche classique qui consiste à optimiser des chemins issus de la planification par des méthodes de lissage est mise en oeuvre. Différentes modifications de la méthode T-RRT et de la méthode de lissage "shortcut" sont présentées. Puis dans une deuxième partie, l'application de STOMP est présentée suivie de sa combinaison à T-RRT .

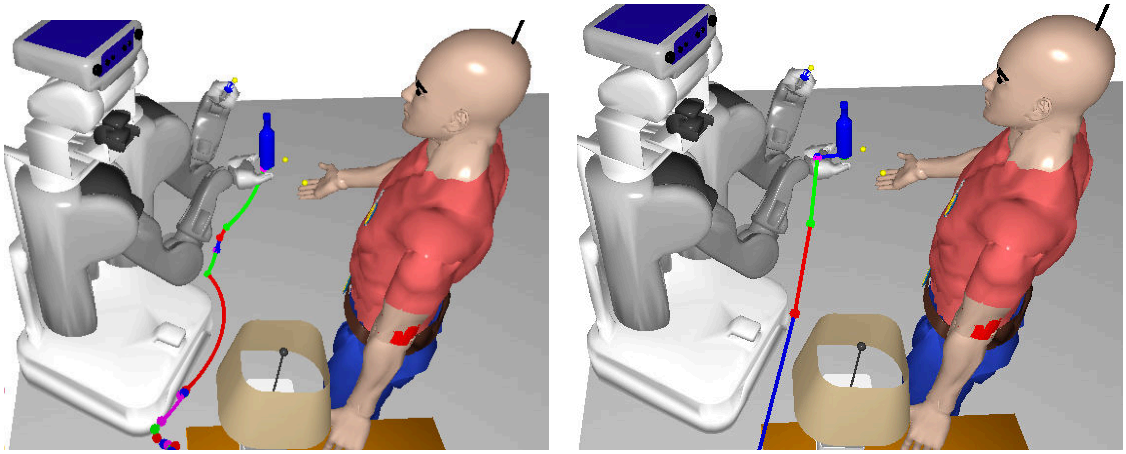
3.4.1 Adaptations de T-RRT

Afin d'adapter T-RRT aux problèmes de manipulation interactive, nous avons tout d'abord étudié l'influence de différents espaces de recherche. En effet, les problèmes de manipulation peuvent être résolus par échantillonnage de l'espace articulaire ou cartésien. De plus, afin d'accélérer la performance de l'algorithme nous proposons une version bidirectionnelle de T-RRT.

Espace de recherche

L'espace des configurations classique d'un bras manipulateur est défini par l'ensemble des degrés de liberté de chaque articulation. Par ailleurs, l'espace cartésien ou espace de l'effecteur considère la situation (position et orientation) de l'effecteur. Ces deux représentations permettent toutes deux de planifier des mouvements pour un bras manipulateur. En effet on peut déduire

la configuration articulaire du bras par cinématique inverse dans le cas cartésien, et obtenir la position et l'orientation de l'effecteur par cinématique directe dans le cas articulaire.



(a) Articulaire (b) Cartésien

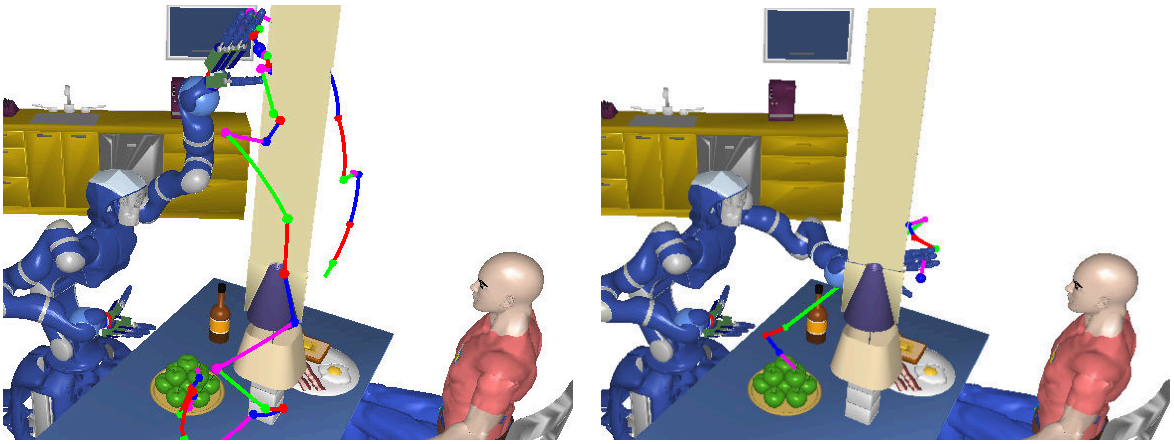
FIGURE 3.8 – Chemin 3D de la pince planifiée dans l'espace articulaire et dans l'espace cartésien.

Pour valider les chemins locaux entre deux configurations lors de la planification dans l'espace cartésien, une fonction est évaluée pour chaque configuration intermédiaire afin de calculer la valeur de DoFs passifs (i.e. les paramètres articulaires). Le pas de discrétisation doit être choisi suffisamment petit pour que le mouvement soit valide. Si la contrainte échoue, le chemin local est considéré comme non valide. Cette technique est souvent utilisée pour planifier avec des contraintes cinématiques, en particulier en présence de chaines fermées [Cortés 05].

Utiliser une représentation cartésienne pour la planification a deux effets. Premièrement, l'interpolation entre deux configurations n'est pas la même (voir Figure 3.8). Deuxièmement le biais de Voronoï associé au RRT conduit à une croissance de l'arbre vers des configurations différentes. La Figure 3.9 illustre deux trajectoires calculées pour le bras gauche du robot Justin. Les configurations initiale et finale sont les mêmes et un obstacle vertical sur la table doit être évité. Le biais vers les régions inexplorées de l'espace de recherche fait croitre le RRT vers des configurations éloignées des butées articulaires ce qui se traduit par la configuration intermédiaire que l'on peut observer sur la Figure 3.9(a). Hormis le fait que ce chemin soit difficile à optimiser par la méthode "shortcut", l'utilisation de l'espace articulaire avec T-RRT s'avère moins efficace dans les problèmes que nous traitons (voir section 4.3).

Ainsi, planifier dans l'espace cartésien peut sembler plus couteux car à chaque étape du RRT (validation des nœuds et des arêtes) la contrainte cinématique doit être validée par un appel à une fonction de calcul de la cinématique inverse. Cependant, ce calcul supplémentaire permet une exploration plus performante. Le biais de Voronoi lors d'une planification dans l'espace cartésien permet de limiter les mouvements de grande amplitude de l'effecteur ce qui est généralement préférable pour la manipulation interactive.

Dans le cas de l'application à T-RRT, les fonctions de coût HRI sont définies dans l'espace cartésien. La fonction de sécurité engendre généralement un coût élevé pour l'effecteur qui se



(a) Articulaire (b) Cartésien

FIGURE 3.9 – Un chemin de manipulation planifié avec un RRT dans un espace de travail contraint par un obstacle vertical. Le chemin calculé dans l'espace des configurations classique engendre des grands mouvements du bras alors que le chemin calculé dans l'espace cartésien est plus direct.

trouve proche de l'humain. Planifier dans l'espace cartésien permet de mieux explorer la carte de coût tout en prenant en compte les contraintes qu'induisent les obstacles. La section 4.3 présente des résultats qui attestent de l'efficacité d'une planification dans l'espace cartésien.

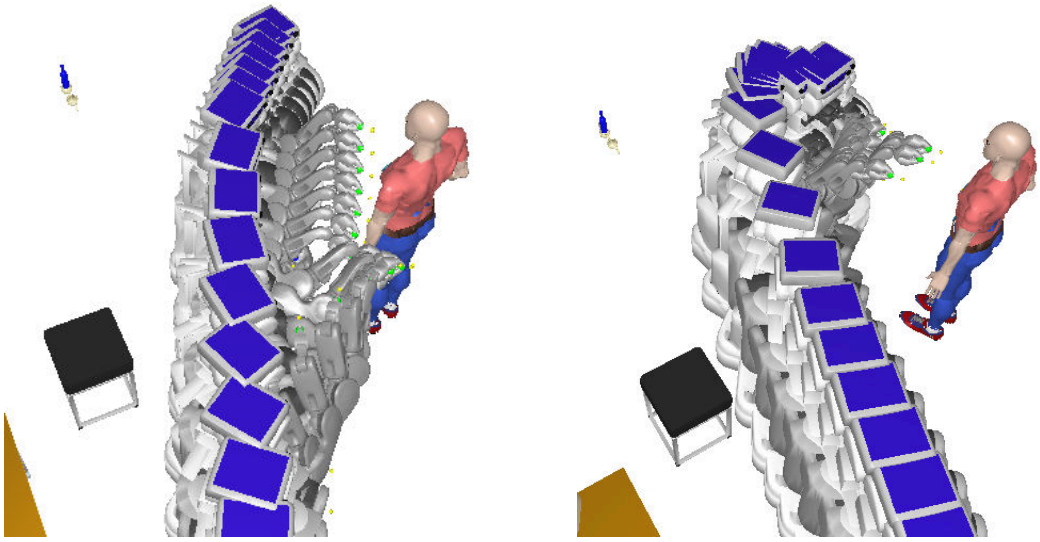
Manipulation et Navigation

Les planificateurs RRT et T-RRT permettent de calculer des chemins de navigation et de manipulations couplées comme illustrés sur la Figure 3.10. Dans ce cas, l'espace des configurations peut comprendre plus de 10 degrés de libertés. Pour traiter ces problèmes il est préférable que les degrés de liberté du bras soient spécifiés dans l'espace articulaire.

En effet, la présence de la base mobile paramétrée par un vecteur $q = (x, y, \theta)$ rend difficile la satisfaction d'une contrainte de cinématique inverse sur le bras manipulateur de manière similaire à la génération de configurations en chaine fermée. Pour ce type de contraintes cinématiques de chaines fermées des techniques ont été mises en oeuvre afin d'échantillonner des configurations valides et leur efficacité a été démontrée pour une utilisation avec PRM et RRT [Cortes 02]. Avec une représentation cartésiennne, les positions de l'effecteur peuvent être échantillonnées loin de la position de la base et provoquer des configurations repliées du bras qui peuvent bloquer la croissance du RRT. De plus il est difficile de générer des configurations respectant ce type de contraintes.

Pas d'expansion

L'algorithme T-RRT repose sur la version "extend" de RRT. Cette version fait croitre des arbres de pas fixes dans l'espace des configurations. La distance entre q_{near} et q_{new} est donc bornée. Le choix du paramètre δ qui contrôle le pas d'expansion doit être choisit suffisamment



(a) RRT (b) T-RRT

FIGURE 3.10 – Les chemins de navigation et manipulation couplé sont calculés avec RRT et T-RRT. (a) Le chemin ne tient pas compte de l'humain et s'en approche de manière dangereuse, (b) le chemin tient compte de l'humain et garde une distance correcte.

petit pour ne pas manquer des pics de coût lors de l'exploration de la carte de coût. Ce pas peut être calculé à partir des dimensions de l'espace de travail. Cependant il peut être important de le faire varier en fonction des degrés de liberté actifs lors de la planification. Un pas trop petit ralentira l'exploration, un pas trop grand provoquera la génération de chemins de mauvaise qualité. Il est choisi à 15 cm pour les problèmes de manipulation avec le robot PR2 pris en compte dans la section 4.3.

Bidirectionnel

L'algorithme de base T-RRT a été développé dans une version mono-directionnelle [Jaillet 10] où un seul arbre est diffusé dans l'espace des configurations enraciné à la configuration initiale. La version standard RRT est souvent utilisée dans une version bidirectionnelle. Dans ce cas, deux arbres sont diffusés simultanément dans l'espace des configurations du robot, l'un enraciné à la configuration initiale et l'autre à la configuration but.

Introduire ce fonctionnement avec T-RRT comporte deux points importants qui doivent être pris en compte dans le test de connexion entre les deux arbres. Les connexions proches doivent être effectuées à une distance suffisamment petite pour ne pas manquer un pic de coût. Les connexions longues doivent considérer le coût des configurations intermédiaires. Ces connexions longues doivent être effectuées avec un seuil maximal, ce qui permet de garantir une exploration minimale de la carte de coût par chacune des composantes. En acceptant uniquement les connexions descendantes, on assure que le travail est minimal pour une des composantes. La méthode de connexion présentée sur les algorithmes 3.5 et 3.6 considère un seuil *maxGap* à partir duquel on considère la descente. Ce seuil permet d'influer sur la rapidité de l'algorithme au détriment de

la qualité des solutions. La section 4.3 présente une évaluation en simulation de cette méthode.

Algorithm 3.5: Test de connexion

```
input  : Les deux arbres T_1, T_2;
         La nouvelle configuration de T_1, q_new;
begin
    q_near ← plusPocheVoisin(q_new , T_2) ;
    if distance( q_new , q_near ) > maxGap then
        return false ;
    if descenteCoût(q_new , q_near) then
        return true ;
    return false ;
end
```

Algorithm 3.6: `descenteCoût`

```
input  : Nouvelle configuration q_new ;
         Configuration de la composante but q_tree;
         Pas d'expansion step;
begin
    LP ← getSegment(q_new , q_tree) ;
    paramMax ← LP.getLength() ;
    p ← step ;
    q_1 ← q_new ;
    while p < paramMax do
        q_2 ← LP.getConfigurationAtParameter(p) ;
        if q_2.cost() > q_1.cost() then
            return false ;
        q_1 ← q_2 ;
        p ← p + step ;
    return true ;
end
```

Test du coût avant les collisions

Le T-RRT effectue deux tests avant l'insertion de chaque configuration q_{new} dans le graphe. Dans la version originale, le test de collision est effectué avant le test de transition qui considère le coût de la configuration. Les performances peuvent être très différentes si on inverse les deux tests. Dans notre cas, la fonction de coût est beaucoup plus rapide à évaluer que la validité du chemin local entre q_{near} et q_{new}. Il est donc judicieux d'effectuer le test de transition avant le test de collision.

3.5 Méthodes locales et optimisation de la solution

Nous avons étudié deux types de méthodes locales dont la méthode "shortcut" largement utilisée et STOMP détaillée dans la section 3.1. Dans un premier temps nous présentons des modifications de la méthode "shortcut" et une extension que nous avons introduite que nous avons nommé "perturbations aléatoires". Elle permet une meilleure optimisation locale en présence de fonctions de coût autre que la distance. Dans un deuxième temps nous proposerons l'adaptation de STOMP aux problèmes de manipulations interactives.

3.5.1 Méthode "shortcut"

La méthode "shortcut" [Berchtold 94, Kavraki 98, Chen 98] consiste à prendre deux configurations q_1 et q_2 puis à tester le chemin local les connectant. Dans le cas où le chemin est plus court et ne présente pas de collision avec l'environnement, la portion du chemin entre q_1 et q_2 est remplacée par le chemin local. Cette méthode peut être adaptée aux cartes de coût en ajoutant un test qui vérifie que le coût du chemin local est moins important que celui de la portion à remplacer. Cette méthode permet de raccourcir un chemin sans altérer sa qualité. L'algorithme 3.7 présente le pseudo code de la version modifiée pour prendre en compte une fonction de coût générale.

Algorithm 3.7: Cost shortcut

```
begin
    while not ConditionArret() do
        (q1, q2) ← P.retourneDeuxConfig();
        LP ←segment(q1, q2) ;
        cost ← P.coutLeLongDeLaPortion(q1, q2) ;
        if estValideEtMinimiseLeCout(LP, q1, q2, cost) then
            P.remplaceLaPortion(LP, q1, q2) ;
end
```

3.5.2 Méthode par perturbations aléatoires

Dans la méthode "shortcut", la recherche est limitée à l'intérieur de l'enveloppe convexe définie par les points du chemin. Cependant un chemin court n'est pas forcément de bonne qualité dans un espace de coût et un détour conduisant à sortir de l'enveloppe convexe initiale peut être préféré afin d'éviter un pic de coût. La Figure 3.11 présente le principe de la méthode de perturbation aléatoire. Une configuration q_{rand} est choisie aléatoirement, puis une configuration $q_{perturb}$ est sélectionnée le long du chemin. La procédure permet de créer un détour qui donne un chemin alternatif qui est évalué puis conservé si son coût est inférieur.

Le pseudo code correspondant est esquissé par l'algorithme 3.8. La première étape consiste à sélectionner une configuration $q_{perturb}$ le long du chemin courant. Cette sélection est biaisée vers les segments de coût plus élevés. Pour cela, les segments de chemin sont triés en fonction

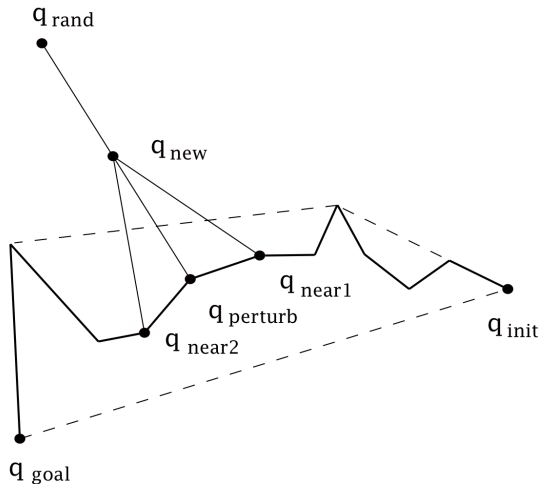


FIGURE 3.11 – La perturbation aléatoire de chemin ne limite pas le nouveau chemin à l'intérieur de l'enveloppe convexe définie par l'ensemble des points du chemin permettant une exploration plus globale du voisinage du chemin

de leur coût et les portions à coût élevé sont sélectionnées avec une probabilité plus importante pour l'optimisation. Dans une deuxième phase, une extension vers une direction aléatoire q_{rand} est effectuée pour sélectionner une configuration q_{new}.

Un paramètre *step* contrôle l'amplitude des perturbations locales. Il est d'abord utilisé pour déterminer la distance entre les configurations q_{near1} et q_{near2}. Il contrôle également la distance entre q_{new} et le chemin, qui est choisie en pourcentage du paramètre *step*. Les valeurs élevées du paramètre *step* tendent à déformer fortement le chemin alors que les valeurs plus faibles ont tendance à affiner localement la solution.

Cette méthode de perturbation explore plus globalement le voisinage du chemin, mais produit généralement des chemins longs. Par conséquent, elle est complémentaire de la méthode de raccourci. Ainsi, dans la phase de post-traitement, il peut être approprié d'utiliser une combinaison des deux méthodes pour obtenir des solutions lisses et de faibles coûts.

3.5.3 STOMP

STOMP (Stochastic Optimization Trajectory for Motion Planning) est une méthode de planification de mouvement décrite dans la section 3.1. L'algorithme génère un ensemble de trajectoires bruitées qui sont ensuite évaluées et combinées pour obtenir une mise à jour de la solution courante. La façon dont les trajectoires sont échantillonnées et combinées permet de garantir que la solution initiale converge vers un minimum local. La trajectoire courante est représentée par un ensemble de points intermédiaires. Pendant le processus d'optimisation le générateur de trajectoires bruitées utilise une matrice de covariance formée grâce à la matrice aux différences finies présentée dans la section 3.1. STOMP cherche en effet à optimiser un coût de collision combiné à un coût associé aux accélérations le long de la trajectoire. La Figure 3.12 présente les trajectoires bruitées produites pour un scénario avec le robot PR2 comprenant les 7 degrés de

Algorithm 3.8: Perturbations Aléatoires

```
input  : Le Chemin P;
output : Le Chemin P;
begin
    while not ConditionArret() do
        q_perturb ← P.configAléatoireSurLeChemin();
        (q_near1, q_near2) ← P.plusProcheVoisin (q_perturb, step);
        q_rand ← tireDirectionAléatoire() ;
        q_new ← expansion(q_perturb, q_rand, step) ;
        LP_1 ←getSegment (q_near1, q_new) ;
        LP_2 ←getSegment (q_new, q_near2) ;
        LP ← LP_1 + LP_2 ;
        if estValideEtDeCoutInférieur(LP, q_near1, q_near2) then
            P.replacePortion (LP, q_near1, q_near2, );
end
```

liberté du bras droit.

Utilisation comme méthode de planification

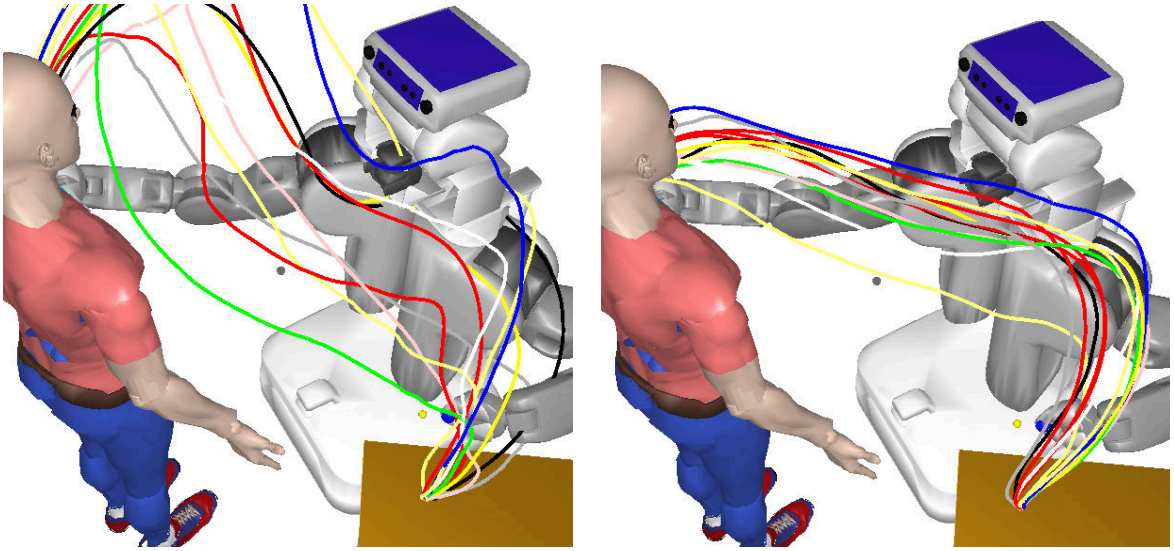
Dans sa version initiale STOMP permet de planifier des chemins sans collisions à partir d'une fonction de coût liée aux obstacles. Le robot est représenté par un ensemble de sphères. L'espace de travail est voxélisé et une transformée de distance euclidienne [Felzenszwalb 04] est utilisée pour estimer la distance aux obstacles. Lors de l'optimisation, les trajectoires bruitées sont évaluées, la distance de chaque configuration intermédiaire à l'obstacle le plus proche est calculée. Afin de prendre en compte les contraintes de distance, visibilité et confort du bras, le coût d'interaction homme-robot est combiné au coût de collision avec une somme pondérée.

Les segments de trajectoire entre les points intermédiaires n'étant pas évalués, il faut donc que le nombre de points soit suffisant pour éviter les collisions avec les obstacles et les auto collisions. Par ailleurs, plus le nombre de points intermédiaires est grand plus chaque itération est coûteuse en calcul. Pour un bras manipulateur comme celui du robot PR2, 100 points sont utilisés pour spécifier la trajectoire.

Paramètres pour STOMP

Hormis les coefficients permettant de pondérer le coût de collision (dans lequel est intégré le coût d'interaction) et le coût lié aux accélérations, l'algorithme comprend deux paramètres principaux. Le premier concerne le nombre de trajectoires qui sont prises en compte à chaque étape et le nombre de trajectoires conservées. Le second concerne l'amplitude du bruit d'exploration (écart type).

En effet, à chaque itération, K trajectoires bruitées sont combinées et les J meilleures sont conservées. Ceci a pour effet de biaiser l'exploration vers des minima déjà atteints. Dans notre implémentation $K = 5$ et $J = 5$. Ces paramètres sont importants et une mauvaise combinaison



(a) Ecart type important (b) Ecart type faible

FIGURE 3.12 – Etape intermédiaire de STOMP avec deux valeurs de l'écart type. 5 trajectoires bruitées sont générées à chaque itérations, les 5 meilleures sont conservées.

de ces paramètres peut empêcher l'algorithme de converger efficacement.

Enfin, le paramètre d'écart type permet de régler l'amplitude du bruit. La Figure 3.12 illustre les trajectoires bruitées avec deux valeurs de l'écart type. Ce paramètre permet de balancer entre une exploration locale et une exploration plus globale autour de la trajectoire. Un écart type trop important ne permet pas de converger efficacement vers un minimum local.

Combinaison de STOMP et T-RRT

T-RRT et STOMP présentent des propriétés complémentaires. T-RRT effectue une exploration globale de l'espace de coût. STOMP optimise localement la trajectoire et présente des propriétés 'anytime'. Dans cette section nous discutons brièvement l'utilisation de STOMP pour optimiser les chemins issus d'une planification T-RRT, comme alternative au lissage "shortcut" ou par déformation qui ne produisent pas de solution "smooth".

Dans sa version initiale STOMP considère la déformation d'une ligne droite dans l'espace articulaire alors que T-RRT produit des chemins en ligne brisée. Afin de surmonter ce problème plusieurs solutions ont été envisagées. Nous avons tout d'abord considéré l'ajout de points intermédiaires au niveau des brisures du chemin T-RRT, puis une modification de l'algorithme STOMP lui même.

En effet, le problème principal se situe au niveau du lissage de l'incrément bruité, effectué par la multiplication des points intermédiaires par la matrice M. Cette étape permet de répartir le gradient sur l'ensemble de la trajectoire grâce à une projection sur la base des vecteurs de R^{-1}. L'application à une ligne brisée empêche la convergence vers une solution lisse. La première solution consiste donc à limiter le coût des accélérations le long de la trajectoire en augmentant le nombre de points aux brisures grâce aux algorithmes introduits dans [Broquère 11]. Cette

solution s'est avérée inefficace en pratique. La deuxième solution consistait à remplacer la multiplication par M par une étape de lissage "shortcut". Cette solution n'a pas permis d'obtenir une convergence vers un minimum local et s'apparente à une exploration aléatoire. Nous avons donc retenu une étape de lissage intermédiaire par "shortcut" qui permet de minimiser les brisures le long de la trajectoire avant son traitement par STOMP. Cette technique permet d'améliorer les chemins issus de la planification T-RRT et d'obtenir des chemins lisses au prix d'un temps de calcul plus long.

3.6 Résultats

Les algorithmes de planification ont été mis en œuvre dans le logiciel de planification de chemin *Move3D* [Simeon 01]. Les descriptions cinématiques de deux robots (PR2 et Rolling Justin) et le modèle d'un humain ont été intégrés afin de représenter des situations de manipulation interactive. Les résultats rapportés dans cette section ont été obtenus sur un processeur 2.6GHz Intel et les valeurs indiquées dans les tableaux correspondent à des moyennes calculées sur 10 passes.

Dans un premier temps, nous analysons la capacité du planificateur introduit dans ce chapitre à produire des chemins qui tiennent compte des contraintes d'interaction homme-robot. Ensuite nous étudions les adaptations de T-RRT, introduites en section 3.1, concernant la version bidirectionnelle et la comparaison entre espace de recherche articulaire et cartésien. Enfin nous comparons les performances de l'algorithme à la méthode RRT* introduite dans l'état de l'art, puis la combinaison de T-RRT avec les extensions des méthodes de post-processing. Finalement, nous analysons la méthode complète qui combine T-RRT et STOMP.

Scenarios

Les trois scénarios utilisés pour les tests, sont présentés sur la Figure 3.13. Sur chaque exemple un chemin en ligne droite entre la configuration initiale et finale est illustré. Les deux exemples avec le PR2 de Willow Garage font intervenir les 7 degrés de liberté du bras droit. L'exemple avec Justin du DLR [Ott 06] considère 3 DoFs supplémentaires pour le torse, faisant donc intervenir 10 DoFs pour la planification. Dans chaque scénario les fonctions de coût présentées dans la section 3.2 sont associées au modèle de l'humain.

Le scénario de la Figure 3.13(a) présente un environnement simple, peu contraint et l'humain est à une distance appropriée. Par contre, dans le scénario de la Figure 3.13(b), l'humain est proche et la table est encombrée ce qui génère des contraintes importantes pour la planification. Dans ce scénario le chemin initial est en collision avec l'humain. Dans le dernier scénario avec le robot Justin, illustré sur la Figure 3.13(c), les obstacles représentés par des lampes doivent être pris en compte pour effectuer un mouvement sans collision.

Ces scénarios sont caractéristiques des situations de manipulation interactive. L'humain demande un objet situé dans l'espace atteignable du robot. Le robot doit effectuer la remise d'objet en minimisant la pénétration dans la zone intime et en restant visible, tout en prenant en compte

les obstacles de l'espace de travail.

3.6.1 Influence de la prise en compte des cartes de coût

Le scénario de la Figure 3.13(c) illustre un problème de planification complexe. Le robot mobile Justin donne un objet à l'humain dans un environnement intérieur encombré notamment par des lampes au plafond. Le problème de planification de mouvement résultant intègre 10 DoFs actifs qui permettent au robot de plier le torse tout en déplaçant le bras afin d'éviter les lampes.

TABLE 3.1 – RRT et T-RRT suivis d'une phase de post-processing sur les cartes de coût élémentaires

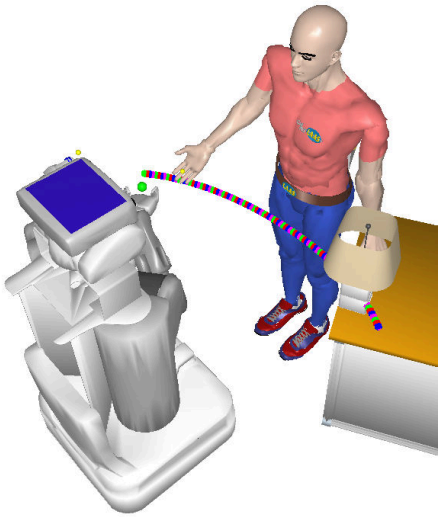
	Time(sec.)	Cost(Before)	Cost(After)
Dist.			
RRT	8.57	212	111
T-RRT	13.31	45	18
Visib.			
RRT	8.64	294	186
T-RRT	7.05	251	176
Reach.			
RRT	8.39	158	89
T-RRT	15.91	117	62

Dans un premier temps (Tableau 3.1), les trois cartes de coûts sont prises en compte séparément lors de la planification de mouvement produisant trois solutions différentes pour le même problème. La Figure 3.14(c) illustre l'effet du critère de distance sur le mouvement du robot. Comme nous pouvons le voir, le chemin résultant pousse le robot loin de l'humain et provoque un comportement plus sûr par rapport au trajet direct généré par une version "standard" de RRT. De même, la Figure 3.14(d) montre l'effet du critère de visibilité, le robot se déplace de manière à ce que l'objet reste aussi visible que possible pour l'homme. Enfin la Figure 3.14(e) montre l'effet du critère d'accessibilité. Le robot se déplace de manière à garder la possibilité à l'homme d'atteindre l'objet d'une manière confortable.

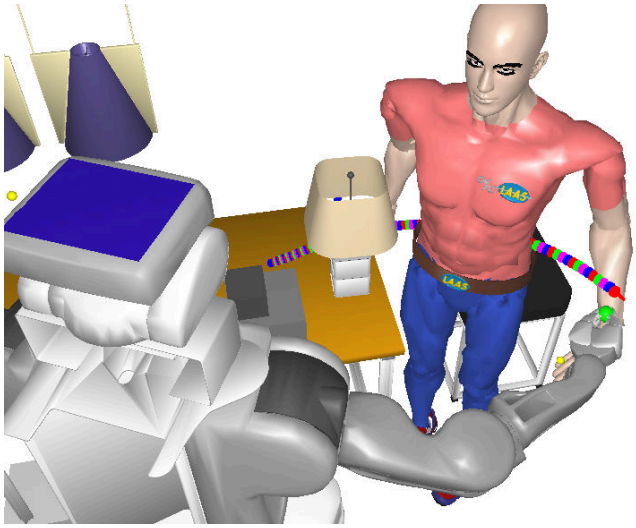
Le temps de planification total inclue le post-traitement. Les coûts des chemins solutions avant et après optimisation, sont comparés sur ces trois cartes de coûts dans le Tableau 3.1. Les phases d'exploration avec un RRT standard et avec un T-RRT sont comparées. Les temps rapportés comprennent un post-traitement de 4 secondes effectué par l'application successive des méthodes de perturbation et de "shortcut". La méthode de perturbation aléatoire permet d'explorer l'espace des configurations plus globalement pouvant générer des chemins bruités alors que "shortcut" permet ensuite de lisser la solution avant son exécution. Un temps égal est alloué aux deux méthodes.

TABLE 3.2 – RRT et T-RRT suivis d'une phase de post-processing sur la combinaison des 3 contraintes

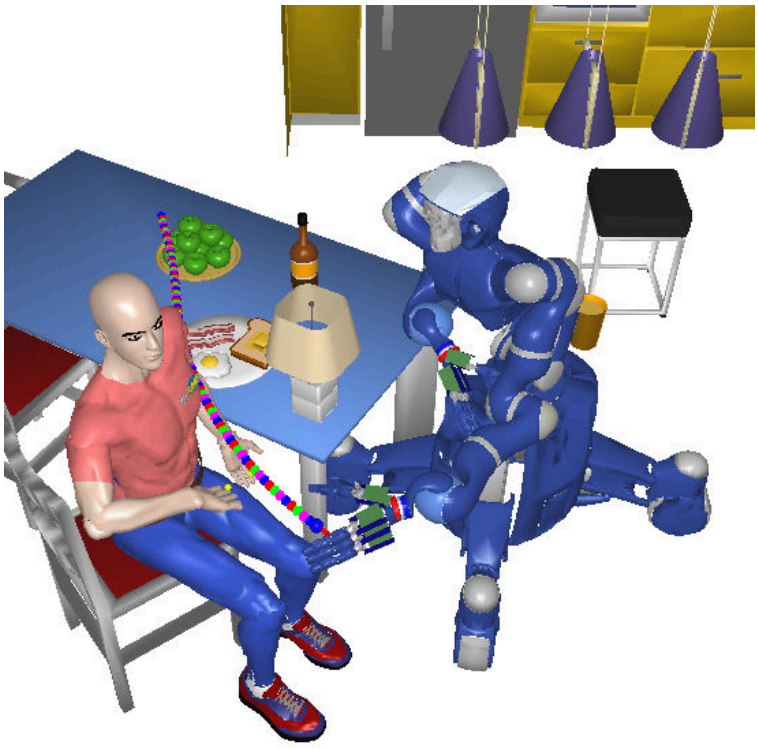
	Time(sec.)	Cost(Before)	Cost(After)
RRT	8.65	179	120
T-RRT	16.69	82	54



(a) PR2 Simple

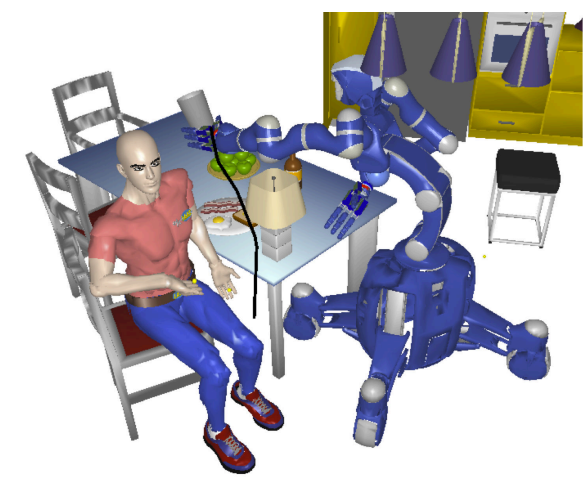


(b) PR2 Contraint

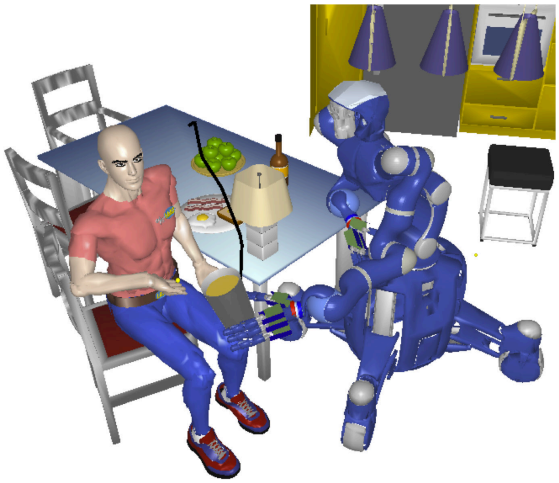


(c) Justin Difficile

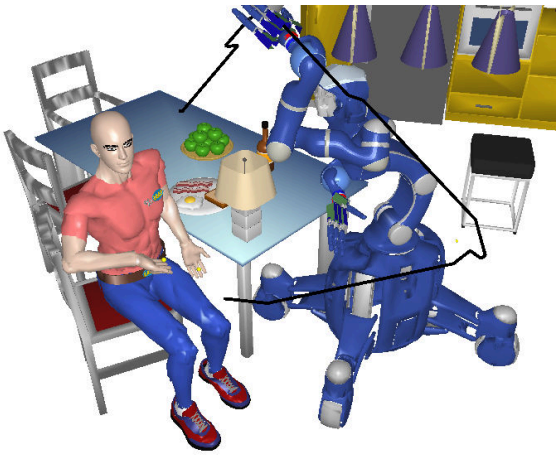
FIGURE 3.13 – Trois scénarios de manipulation en interaction avec l'homme. La trajectoire en ligne droite entre la configuration initiale et la configuration finale est donnée par les points de couleur. La configuration courante du robot correspond à la configuration finale. Pour l'exemple contraint avec le robot PR2 le chemin initial est en collision avec l'humain. En plus des 7Dof actifs dans les deux premiers scénarios avec le PR2 (a) et (b), le scénario avec Justin comprend 3Dof actifs supplémentaires pour le torse.



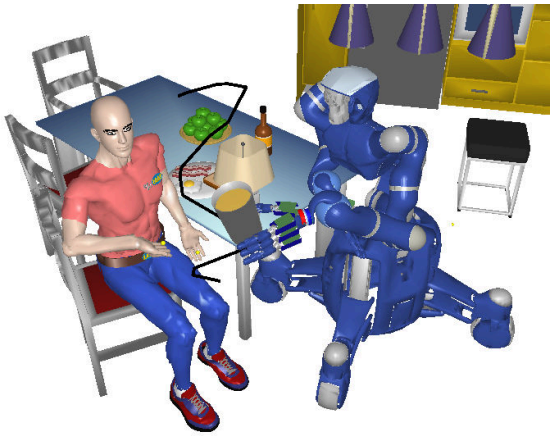
(a) Configuration Initiale



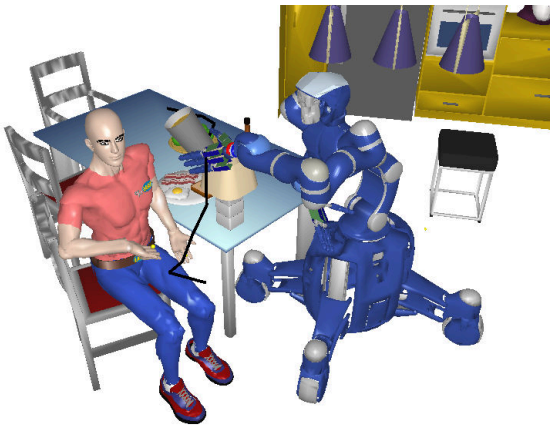
(b) Configuration Finale



(c) Coût de distance

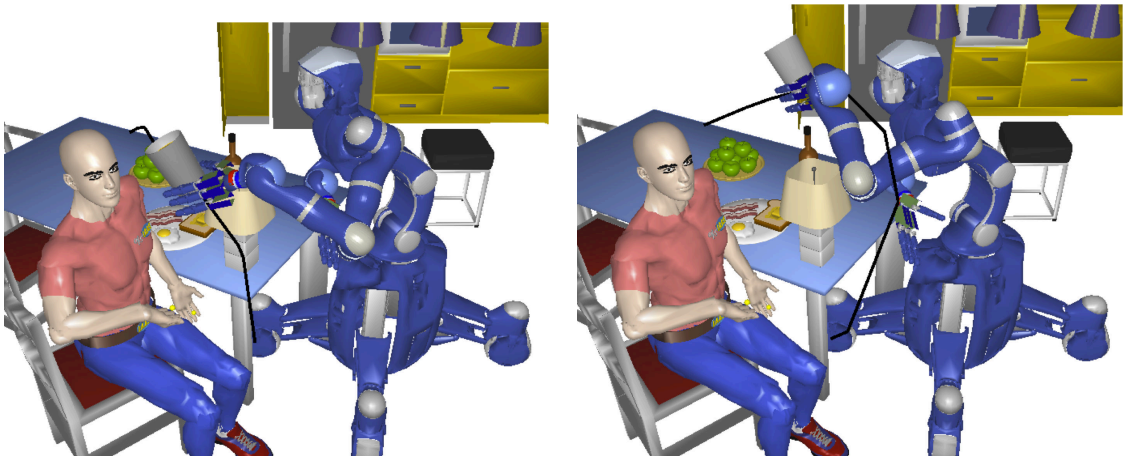


(d) Coût de vislibilité



(e) coût de "confort du bras"

FIGURE 3.14 – Les trois fonctions de coût conduisent à trois espaces de coût qui peuvent être pris séparément en entrée du planificateur basé sur T-RRT suivis d'une phase de post-processing.



(a) Chemin RRT (b) Chemin Costmap

FIGURE 3.15 – L'homme est assis sur une chaise et détourne le regard du robot. Le mouvement trouvé par un planificateur "standard" ne traite pas la présence de l'homme. Par contre le mouvement calculé en prenant en compte des trois contraintes génère un mouvement plus confortable. En suivant ce chemin, le robot reste aussi visible que possible à une distance rassurante tout en laissant la possibilité à l'homme de saisir l'objet.

Le planificateur T-RRT améliore considérablement le coût des solutions sur la carte de coût de distance parce que le chemin initial RRT est de très mauvaise qualité, étant très proche de l'humain. Pour les contraintes de visibilité et d'accessibilité, les solutions sont plus semblables à la solution RRT standard de la Figure 3.22(a). C'est ce qui explique le gain moins important trouvé par la planification dans l'espace de coût du Tableau 3.1.

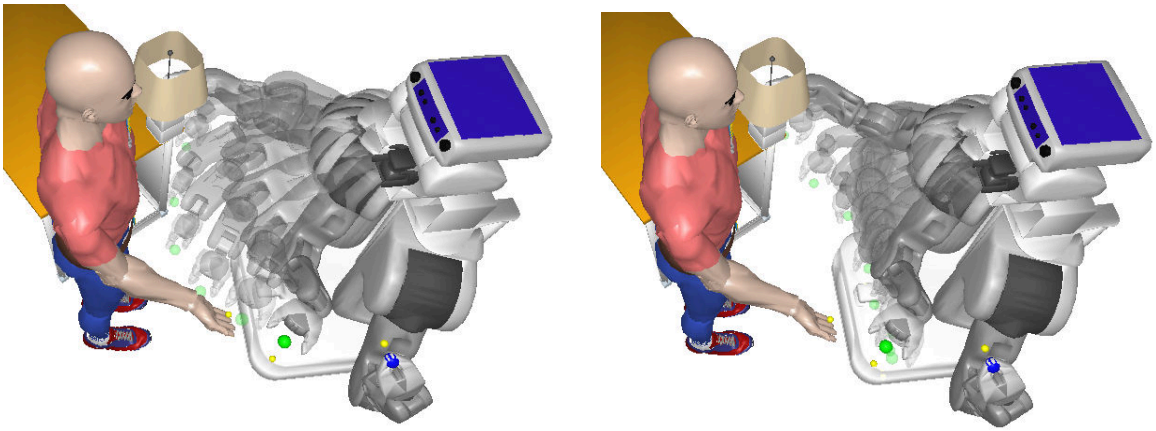
Enfin, le Tableau 3.2 rapporte les résultats obtenus lors de l'examen de la combinaison des trois cartes de coût. La solution obtenue réalise un bon compromis entre les trois contraintes. Ceci est illustré sur la Figure 3.15.

3.6.2 Etude de T-RRT

bidirectionel

Le tableau 3.3 compare une version classique de RRT avec deux versions de T-RRT, l'une bidirectionnelle et l'autre monodirectionnelle dans les trois problèmes de la Figure 3.13. Les deux versions de l'algorithme T-RRT utilisent une valeur de $nFailMax = 10$, ce qui correspond à une version "greedy" de l'algorithme.

La Figure 3.16 présente deux chemins calculés pour l'exemple "simple" avec RRT et T-RRT. Le chemin calculé par RRT ne prend pas en compte la présence de l'humain et pénètre dans la zone intime pouvant le mettre en danger. Au contraire le chemin calculé avec T-RRT qui prend en compte une combinaison des contraintes et maintient un dégagement important avec l'humain. Sur cet exemple, bi-T-RRT génère environ 10 fois plus d'expansions que bi-RRT avant de trouver une solution, ce qui est normal car T-RRT tient compte de la carte de coût et traite donc un problème plus difficile. La version monodirectionelle de T-RRT, elle, génère approximativement



(a) RRT (b) T-RRT

FIGURE 3.16 – Un chemin RRT et T-RRT calculé avec une version bidirectionnelle des deux algorithmes sur l'exemple de la Figure 3.13(a).

100 fois plus d'expansions et s'avère être 400 fois plus lente que la version bidirectionnelle de T-RRT. Notons que cette tendance est amplifiée sur les deux autres exemples sur lesquels elle ne permet pas de trouver de solution dans un temps raisonnable.

Ceci est dû à la présence d'un "passage étroit" en coût. En effet si la configuration but se situe derrière un col de coût, l'algorithme doit explorer la région de bas coût à proximité de la configuration initiale avant de le franchir. La version bi-drectionelle de T-RRT permet de faire croitre un arbre enraciné à la configuration but qui est à l'intérieur de la zone intime avec un coût HRI élevé. La barrière de coût formée par la zone intime n'a donc pas à être franchie. Sur cet exemple, nous pouvons remarquer que la qualité de la solution est supérieure avec un gain en temps, en nombre d'expansions et en nombre de noeuds dans le graphe.

TABLE 3.3 – Comparaison T-RRT et bi-T-RRT sur les exemples de la Figure 3.13

	Time(sec.)	Cost	Nodes	Expansions
Simple				
bi-RRT	0.38	65421	13	17
bi-T-RRT_{10}	0.53	450	37	234
T-RRT_{10}	200.31	461	4907	19376
Constraint				
bi-RRT	0.87	2240000	77	275
bi-T-RRT_{10}	1.72	57199	206	1183
T-RRT_{10}	x	x	x	x
Justin				
bi-RRT	2.29	19942	130	342
bi-T-RRT_{10}	6.10	1451	279	1527
T-RRT_{10}	x	x	x	x

Espace articulaire et espace cartésien

Les deux espaces de recherche sont comparés sur les trois problèmes de la Figure 3.13. Un T-RRT bidirectionnel est utilisé avec valeur de $nFailMax = 30$, ce qui correspond à une valeur intermédiaire de l'algorithme. Les temps de calcul et la qualité des solutions sont reportés sur le tableau 3.4. Nous pouvons constater que pour un scénario simple, l'espace de recherche cartésien permet de trouver des solutions de meilleure qualité en un temps inférieur à celle trouvée en utilisant l'espace articulaire. Cette observation se renforce sur les scénarios contraints (PR2 et Justin). Notons, pour l'exemple simple, que la planification dans l'espace cartésien génère plus d'expansion pour un nombre de noeuds inférieur. En effet lors de l'expansion, la contrainte cinématique est évaluée et peut invalider l'expansion ce qui tend à générer plus d'expansions pour résoudre des problèmes peu contraints. Nous pouvons aussi remarquer que pour le scénario plus dimensionné et plus contraint dans le cas du robot Justin illustré Figure 3.13(c) , la version articulaire de l'espace de recherche ne permet pas de trouver de solution dans un temps raisonnable.

TABLE 3.4 – Espace articulaire et espace cartésien sur les exemples de la Figure 3.13

	Time(sec.)	Cost	Nodes	Expansions
Articulaire (Simple)	0.60	418	89	825
Cartésien	0.52	222	65	1215
Articulaire (Contraint)	3.34	1157	347	3416
Cartésien	1.61	743	154	2764
Articulaire (Justin)	x	x	x	x
Cartésien	9.04	1345	366	4457

T-RRT et RRT*

RRT* [Karaman 11] est une méthode de planification basée sur RRT qui présente des propriétés de convergence vers le chemin optimal. L'algorithme RRT* permet d'optimiser n'importe quel coût le long de lu chemin pouvant s'exprimer par l'intégrale d'un paramètre positif (e.g. longueur ou un coût le long du chemin). Il présente également des propriétés "anytime" améliorant la solution de manière monotone pendant la planification. Dans cette section, nous présentons une analyse comparative entre T-RRT et RRT* sur l'exemple "simple" présenté sur la Figure 3.13(a) qui montre la meilleure performance de T-RRT.

Le tableau 3.5 présente les résultats de la version bidirectionnelle de T-RRT pour trois valeurs différentes du paramètre $nFailMax$ $(10, 30, 100)$ et deux valeurs intermédiaires de la convergence de RRT*. La convergence du coût de la solution courante obtenue avec RRT* est présentée sur la Figure 3.17 pour 90 secondes intégrant une moyenne de 10 passes. Le tableau 3.5 présente également le temps initial auquel un chemin de faible qualité a été trouvé (2.72sec) avant d'être amélioré par des expansions lourdes. Différentes métriques de qualité comme le coût maximal ou le coût mécanique présenté dans [Jaillet 10] sont également reportées.

RRT* nécessite 4.53 fois le temps de la version intermédiaire ($nFailMax = 30$) de T-RRT pour trouver une solution de moins bonne qualité. T-RRT explore donc la carte de coût de

TABLE 3.5 – Comparaison de T-RRT et RRT* sur l'exemple de la Figure 3.13(a). Différentes valeurs de $nFailMax$ (10, 30, 100) sont utilisées pour T-RRT et les deux valeurs de RRT* sont prises à 20 sec puis 1800 sec

	Time(sec.)	Extensions	Nodes	Max	Mecha	Integral
bi-T-RRT_{10}	0.42	229	37	175	119	435
bi-T-RRT_{30}	0.60	508	42	117	58	377
bi-T-RRT_{100}	0.58	808	38	108	33	358
RRT^*	2.72	x	x	301	235	434
RRT^*	20	264	192	285	215	335
RRT^*	1800	12161	7563	235	196	282

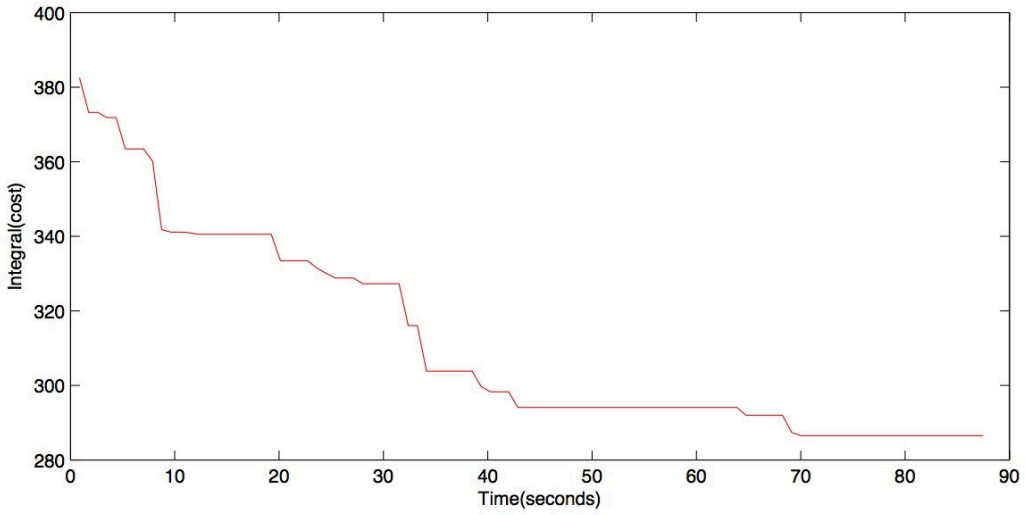


FIGURE 3.17 – Convergence de l'intégrale de coût de RRT* sur l'exemple de la Figure 3.13(a)

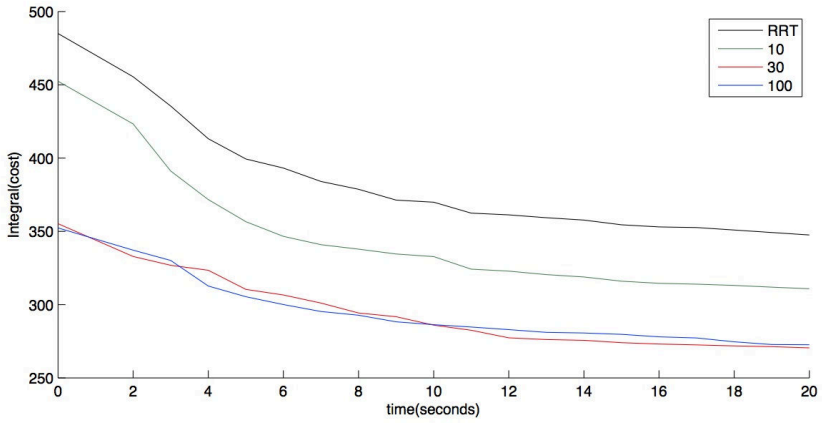
manière plus efficace en produisant rapidement un chemin qui ne pénètre pas dans la zone intime de l'homme. Cette différence de performance vient du test de transition sous-jacent à T-RRT qui rejette les échantillons et force l'algorithme à conserver peu de noeuds dans l'arbre, 38 noeuds pour 808 expansions pour une valeur de $nFailMax = 100$. RRT* produit en 20 secondes 4 fois moins d'extensions que T-RRT en une seconde. Cependant, T-RRT minimise initialement le travail mécanique le long du chemin [Jaillet 10], ce que l'on peut observer sur le tableau 3.5. Il ne permet donc pas d'optimiser n'importe quelle métrique additive comme RRT* qui converge vers une solution minimisant l'intégrale de coût en 1800 secondes.

3.6.3 Application de "shortcut" et de la méthode de perturbation

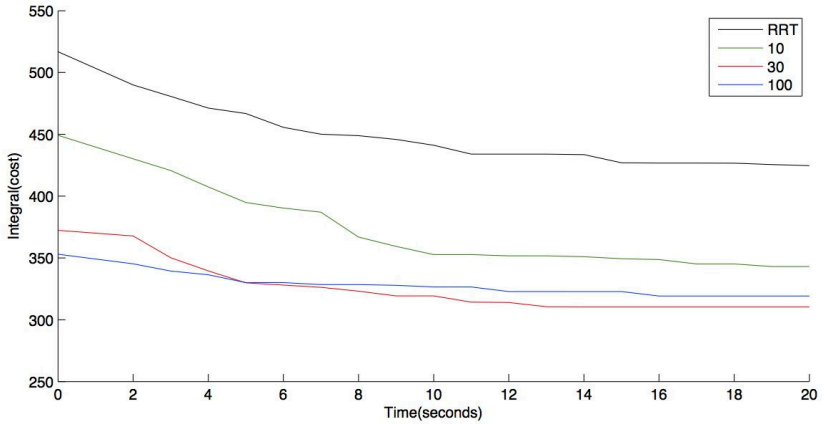
La Figure 3.18(a) présente des courbes de convergences pour la méthode "shorcut" et la méthode de perturbation présentée en section 3.4 sur l'exemple de la Figure 3.13(a). Les méthodes locales sont appliquées à des solutions RRT et T-RRT avec différentes valeurs du paramètre $nFailMax$ (10, 30, 100). Nous pouvons observer que la méthode de déformation est sensiblement

plus lente, notamment entre 0 et 4 secondes. Ceci est dû aux opérations de déviation qui sont plus lourdes que les opérations de "shorcut".

Les méthodes locales seules ne permettent pas d'atteindre les qualités obtenues en les combinant à des versions plus sélectives de T-RRT. Par ailleurs, la combinaison de T-RRT aux méthodes d'optimisation locales (coût inférieur à 300 avec "shortcut") est plus performante que RRT* sur 20 secondes (coût de 335).



(a) Convergence "shortcut"



(b) Convergence perturbation

FIGURE 3.18 – Convergence de la méthode "shortcut" pour différentes qualités des chemins initiaux (RRT et T-RRT avec des valeurs du paramètre $nFailMax$ 10, 30 et 100).

3.6.4 Planification avec STOMP

STOMP, introduit en section 3.1, peut être utilisée pour résoudre le problème de planification de mouvement directement. En effet dans la version initiale de l'algorithme, la planification démarre avec une ligne droite dans l'espace articulaire échantillonnée sur 100 points représentées sur les Figures 3.13(a) et 3.13(b). Dans cette section, nous étudions la résolution d'un problème de planification avec des contraintes d'interaction homme-robot par STOMP. Dans la section

suivante, nous analyserons sa combinaison avec T-RRT.

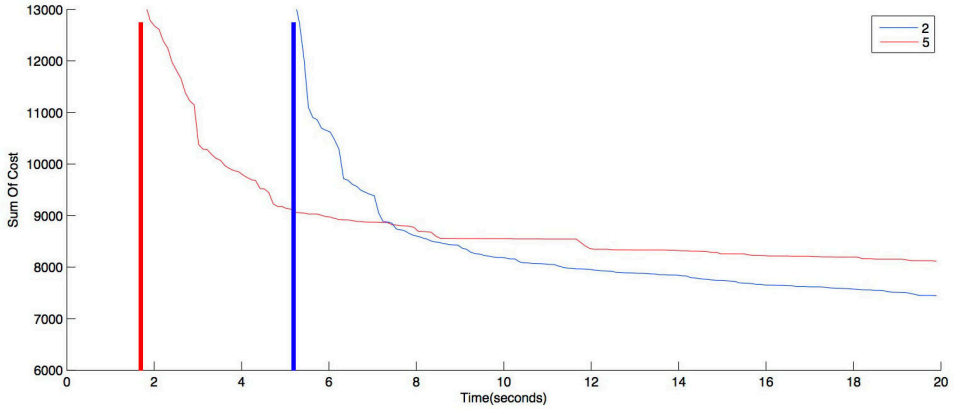
L'écart type est un paramètre important de STOMP qui vient directement multiplier le bruit ajouté à la trajectoire courante afin d'obtenir une exploration plus ou moins locale. La Figure 3.19 présente la convergence de la somme des coûts des 100 points intermédiaires pour deux valeurs du paramètre d'écart type, et sur les deux scénarios "simple" et "contraint" de la Figure 3.13. Pour le scénario "simple", les trajectoires sont sans collision dans 100% des cas, avec un temps moyen de convergence de 5.21 sec pour un écart type faible et 1.74 sec pour un écart type important. Pour le scénario "contraint", seulement 50% des cas sont sans collision après 20 sec avec un écart type important et deviennent sans collision au bout de 6.17 secondes en moyenne alors que 70% des cas sont sans collision pour l'écart type faible au bout de 9.17 secondes en moyenne. Les courbes de convergence (Figure 3.19) du scénario "simple" sont donc mises à l'échelle pour observer le phénomène de sortie de collision qui n'apparait pas sur le scénario "contraint".

L'optimisation avec un écart type important permet donc de sortir plus rapidement de collision dans le cas "simple". Mais un écart type plus petit permet de converger vers des solutions à plus bas coût, notamment pour le scénario moins contraint. Ce résultat est cohérent, il illustre le fait qu'une exploration moins locale permet de trouver plus de configurations sans collisions dans ces environnements. Cependant, il est difficile de généraliser ce résultat à l'ensemble des problèmes que l'ont peut rencontrer. En effet, dans le cas d'une trajectoire initiale très peu en collision dans un espace encombré, une déformation très locale permettra de sortir de collision alors qu'une déformation plus importante peut ne pas y parvenir.

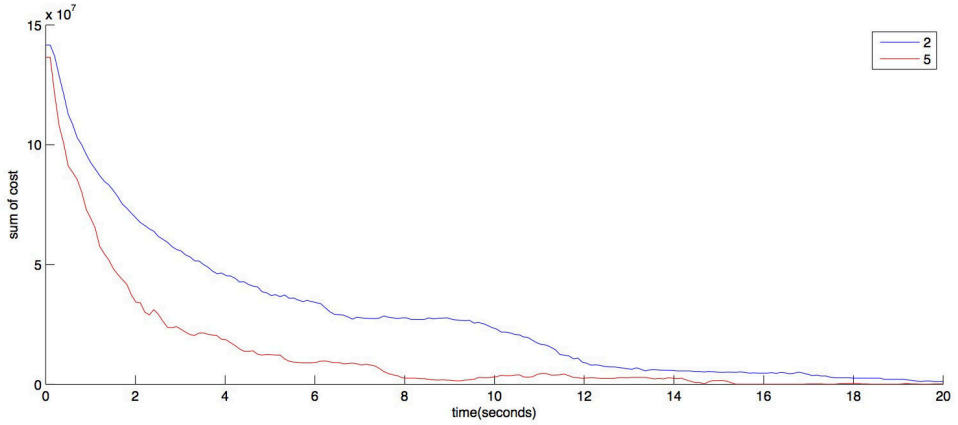
Les chemins produis par STOMP sont propices à la manipulation interactive de part leur nature lisse. De plus STOMP ne requiert pas de phase d'optimisation contrairement aux méthodes traditionnelles de planification par échantillonnage. Il est envisageable de l'utiliser comme méthode de planification dans le cas des scénarios peu contraints comme le scénarios "simple" où elle permet d'obtenir des chemins de bonne qualité en quelques secondes (2 à 6 sec). Cependant, l'analyse du deuxième exemple montre l'incapacité de STOMP à trouver des chemins sans collision dans des environnements contraints. La combinaison à une méthode d'exploration globale comme T-RRT qui produit rapidement des chemins sans collision de bonne qualité est donc souhaitable.

3.6.5 Combinaison de STOMP et de T-RRT

Nous avons proposé dans la Section 3.4 une solution pour combiner T-RRT et STOMP. Elle consiste à introduire une phase de lissage du chemin entre la planification et l'optimisation. La convergence de la méthode "shortcut" sur les deux scénarios "simple" et "contraint" avec le robot PR2 est présentée sur la Figure 3.20. Les courbes sont similaires pour les deux scénarios. Il est donc raisonnable de fixer un intervalle de temps pour lisser la solution initiale. La Figure 3.22, présente deux courbes de convergence de l'optimisation STOMP avec un écart type faible à partir de solutions T-RRT lissées par une phase de "shortcut" de 1, 3 et 5 secondes . Nous pouvons observer que la durée de lissage importe peu sur la performance de STOMP qui atteint un minium local en 10 secondes sur les deux scénarios.



(a) Convergence sur l'exemple PR2 simple



(b) Convergence sur l'exemple PR2 contraint

FIGURE 3.19 – Courbes de convergence de STOMP sur 20 secondes pour deux valeurs du paramètre d'écart type sur les deux exemples avec le robot PR2 de la Figure 3.13. Dans le cas de "simple", les barres de couleur rouge et bleue indiquent la sortie de collision et les courbes sont mises à l'échelle afin d'observer la phase d'optimisation.

Dans ce cas, des solutions sans collision minimisant les coûts HRI sont trouvées dans 100% cas. La Figure 3.22 présente 3 trajectoires sur le scénario "contraint" issues de la planification RRT, T-RRT et de la combinaison de T-RRT et STOMP. Nous pouvons observer que la trajectoire combinant les deux méthodes optimise mieux les contraintes d'interaction homme-robot en se dégageant au maximum de l'humain. De plus, STOMP de par son caractère "anytime" permet de prendre en compte plus efficacement des contraintes HRI dans un contexte de replanification dynamique.

3.7 Conclusion

Nous avons présenté une nouvelle approche pour la planification de mouvement en interaction avec l'homme. Ces techniques basées sur une représentation de l'interaction sous forme de carte de coût étaient limitées à des environnements peu contraints. Le passage à des cartes de coût

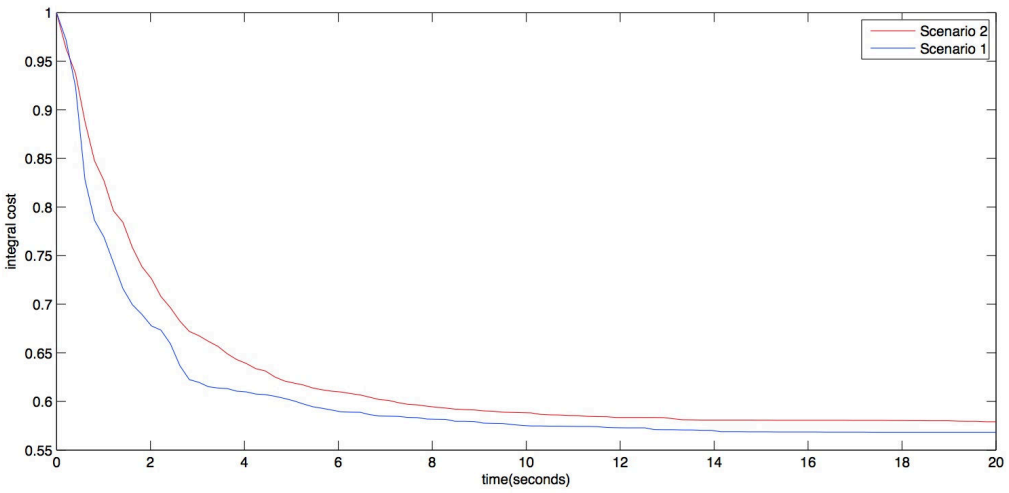
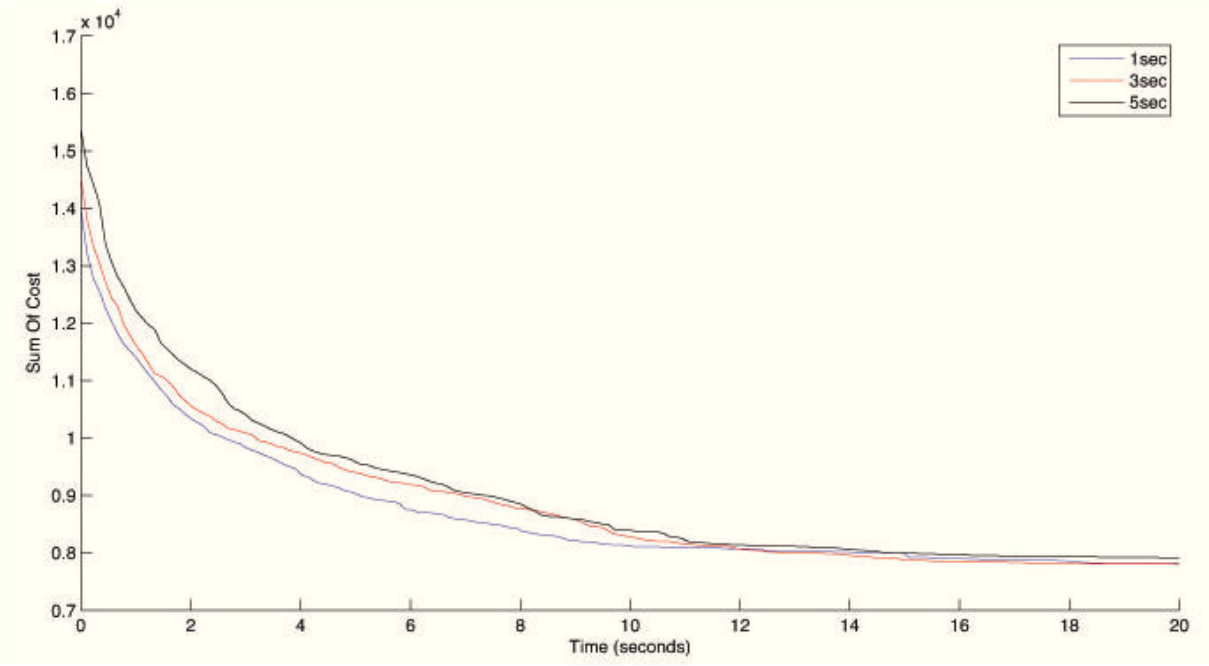


FIGURE 3.20 – Méthode "sortcut" sur les deux scénarios de la Figure 3.13 après T-RRT avec $nFailMax$ de 30. La valeur de coût est normalisée.

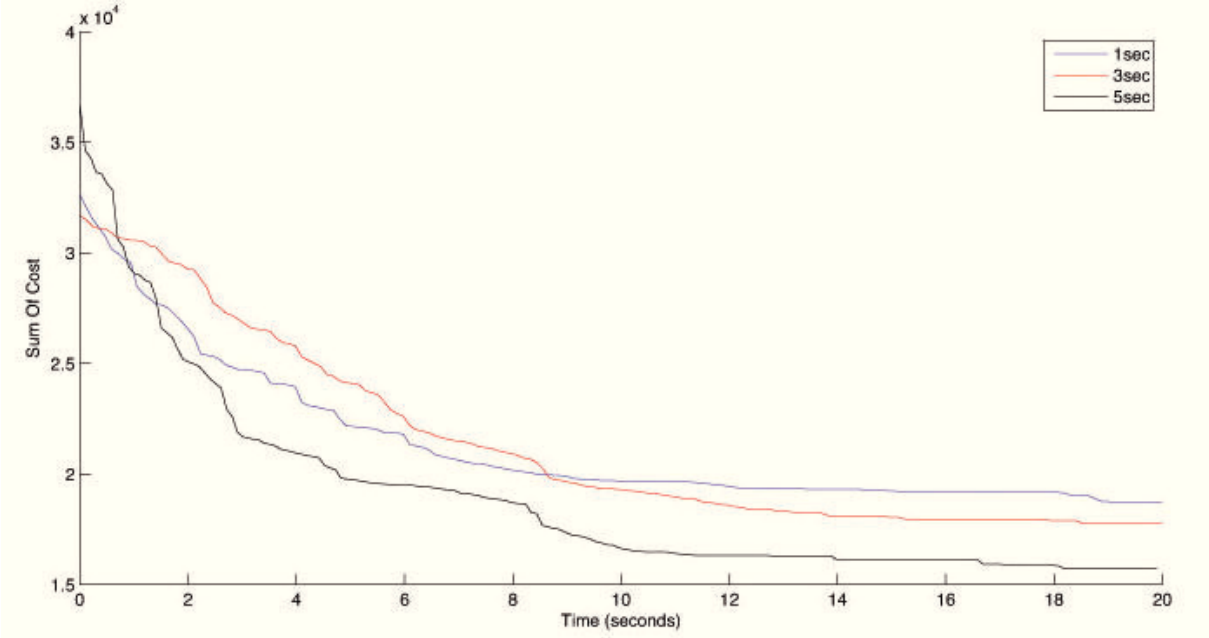
définies dans l'espace des configurations du robot était donc nécessaire et a permis l'application de techniques de planification récentes.

Ces techniques basées sur la planification de mouvement par échantillonnage et l'optimisation stochastique ont été appliquées pour traiter ces problèmes. Nous avons montré les adaptations nécessaires de T-RRT, des méthodes de lissage et de STOMP afin de traiter ces problèmes de manipulation interactive.

L'algorithme STOMP a été adapté et évalué pour planifier des mouvements de manipulation directement, ce qui s'est avéré inefficace pour traiter des problèmes contraints. Par la suite, une connexion avec T-RRT a été possible en utilisant une phase intermédiaire de lissage du chemin. Les chemins obtenus par ce procédé permettent de satisfaire différentes métriques telles que la somme des coûts le long du chemin, le maximum et l'intégrale des coûts. Les chemins obtenus sont lisses et le caractère "anytime" de STOMP permet d'envisager la prise en compte de changements dans l'espace de travail.

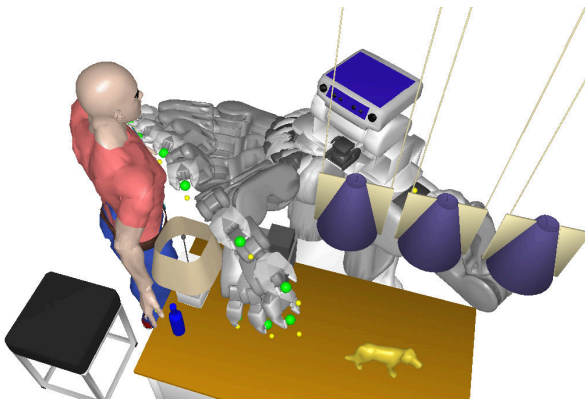


(a) Convergence sur l'exemple PR2 simple

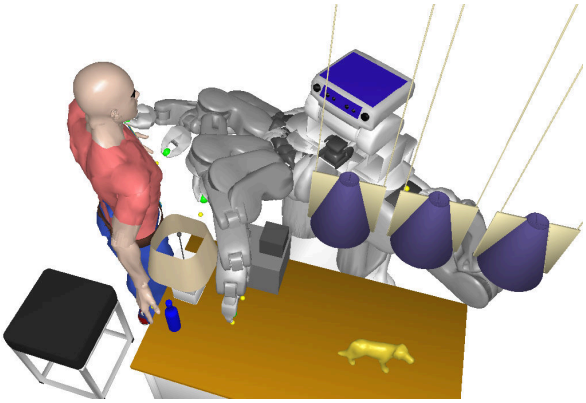


(b) Convergence sur l'exemple PR2 contraint

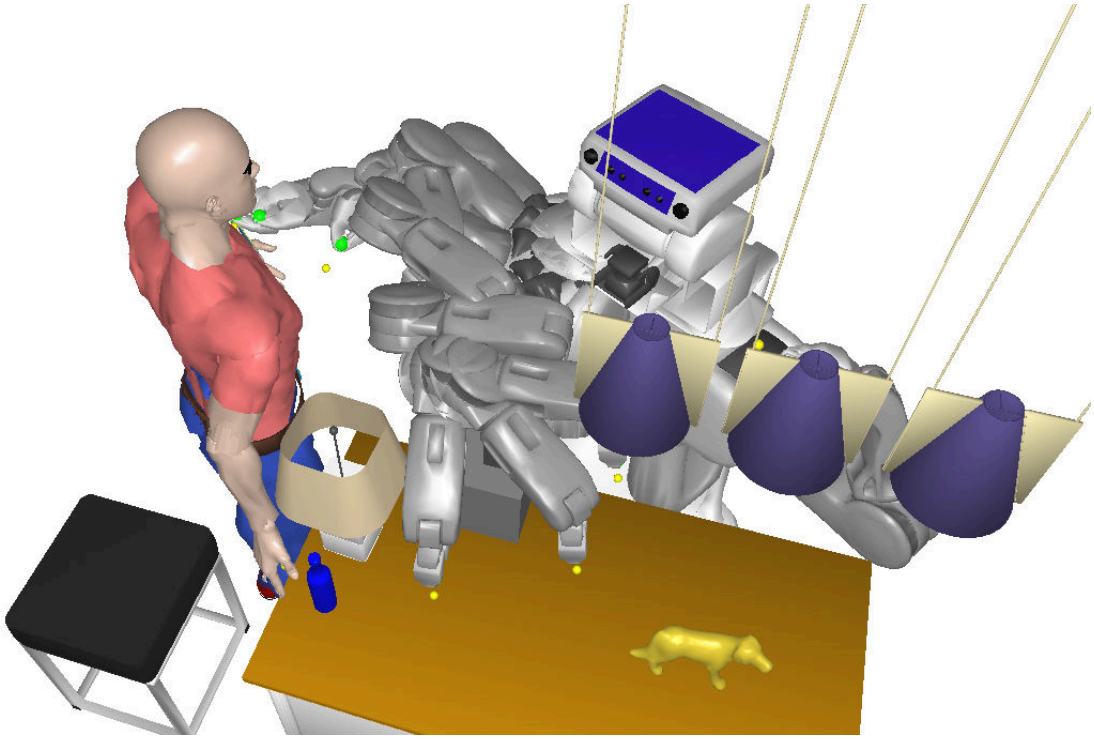
FIGURE 3.21 – Convergence de la somme des coûts de STOMP après T-RRT pour trois temps de lissage sur les deux exemples de la figure 3.13



(a) RRT



(b) T-RRT



(c) T-RRT et STOMP

FIGURE 3.22 – Trajectoires calculées avec différentes méthodes de planification. La planification RRT passe très proche de l'humain de manière dangereuse. La planification T-RRT prend en compte les critères de sécurité. La planification combinant T-RRT et STOMP permet d'obtenir un mouvement lisse et sûr dans un environnement contraint.

4

Partage de l'effort dans les tâches de transfert d'objet

Dans ce chapitre, nous étudions le transfert d'objet du robot à l'homme. Nous considérons les cas pour lesquels le robot ne peut pas directement accéder à la personne ou ceux pour lesquels un petit déplacement de l'humain permet d'accélérer le transfert. Dans ce type de situations des stratégies d'échange comme à travers une fenêtre ou une clôture sont préférables. Ainsi, l'homme "partage l'effort" avec le robot car il est amené à se déplacer pour la réalisation de la tâche. Il est clair qu'il peut y avoir plusieurs solutions plus ou moins satisfaisantes en terme de confort ou de temps pour atteindre l'objet. Nous pouvons observer que la capacité de déplacement de l'humain ne doit être utilisée que lorsque cela entraîne une réduction importante de la durée de la tâche. Cette idée est à la base de la contribution de [Shah 11] qui cherche la performance d'équipe humain-robot.

Dans ce chapitre, nous proposons de traiter ce problème comme une instance particulière du problème de planification de mouvement [Latombe 91b, Choset 05b, LaValle 06]. Nous avons proposé une formulation dans laquelle les mouvements de l'homme sont intégrés afin de tenir compte de son implication dans la tâche.

Afin de traiter ces problèmes d'échange d'objet qui nécessitent une navigation de la part du robot nous avons développé un algorithme qui consiste à chercher une configuration de transfert d'objet dans l'espace de travail tout en planifiant les mouvements d'approche du robot et de l'homme. Cette technique est la première à prendre en compte cette notion d'effort partagé entre les mouvements du robot et de l'homme permettant d'accéder à la position de transfert. Notre objectif étant de prendre en compte l'*empressement* d'obtenir l'objet ou les *capacités physiques* de l'humain, nous introduisons un ensemble de critères pour que l'échange soit sûr, lisible et fluide. Ces critères sont inspirés des approches basées sur des coûts, introduites dans les

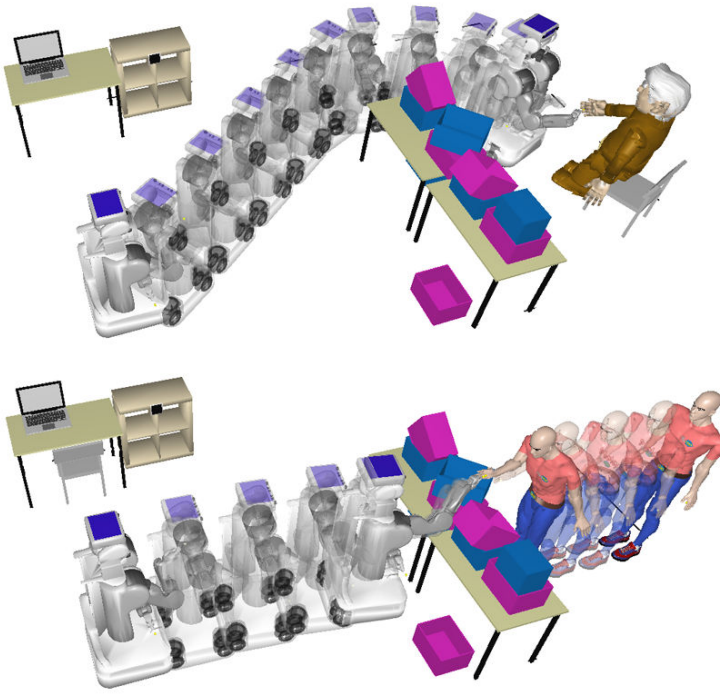


FIGURE 4.1 – Une personne jeune et pressée d'obtenir un objet exprimera plus de confort en l'obtenant par dessus une table encombrée alors qu'une personne âgée préférera obtenir l'objet en restant assise.

chapitres précédents, qui tiennent compte des préférences de l'homme en accord avec la théorie de la *proxémie* [Hall 63]. Ce type d'approche pour la génération de mouvement en interaction avec l'homme est un sujet de recherche très actif [Kulic 07b, Sisbot 07a, Lam 10, Scandolo 11a]. Nous introduisons également un nouveau critère nommé *mobilité* afin d'équilibrer "l'effort partagé" et le confort des plans produits. La solution algorithmique développée est simple et efficace. La Figure 4.1 présente l'exemple d'une tâche de transfert résolue par notre planificateur avec des réglages différents du paramètre de *mobilité*. Après avoir introduit le problème de manière formelle ainsi qu'un algorithme permettant de le traiter, nous présentons les résultats d'une étude HRI effectuée sur un robot PR2 au LAAS qui démontre la pertinence de l'approche.

4.1 Le problème du partage d'effort dans le transfert d'objet homme-robot

Dans cette section, nous proposons une définition formelle du problème de planification d'un transfert d'objet. Nous présentons d'abord les entrées et sorties du problème, puis l'espace de recherche et les contraintes de faisabilité et d'interaction qui doivent être prises en compte pour générer des solutions efficaces et agréables.

4.1.1 Entrées et sorties

Pour tenir compte du mouvement de l'humain nous considérons des plans composés de deux chemins : celui du robot et celui de l'humain. Ces chemins spécifient le mouvement de la posture initiale à la posture but de transfert. Nous postulons qu'il est essentiel de tenir compte du chemin de l'humain parce que l'espace de travail peut être encombré, formant des passages étroits ou des barrières empêchant le robot d'accéder à l'humain. Par ailleurs, dans certains cas, le fait que l'homme puisse se déplacer vers le robot peut se traduire par un gain important en terme de temps d'exécution. Dans ces cas, la maîtrise de la *mobilité* du receveur humain s'avère essentielle. Cette notion de mobilité du récepteur humain est analysée dans l'expérience HRI à la fin du chapitre.

Formellement les entrées du problème sont les configurations initiales du robot q_r^{init} et de l'humain q_h^{init}. Le problème prend également en entrée leurs modèles cinématiques et la représentation de l'espace de travail W. Le plan résultant est composé des deux chemins τ_r et τ_h représentés par des courbes paramétrées dans l'espace des configurations du robot et de l'humain. L'espace des configurations possibles de transfert d'objet dans l'environnement est introduit dans la section 4.1.2 dans laquelle nous présentons un ensemble de propriétés qui contraignent la faisabilité et la qualité du plan au regard de l'interaction entre l'humain et le robot. Dans cette section nous donnons une définition formelle de ces propriétés qui permettent de produire des plans où l'effort est partagé tout en tenant compte du confort de l'humain.

4.1.2 L'espace des configurations de transfert

Nous considérons l'espace des configurations formé par le produit cartésien entre l'espace des configurations du robot C_r et l'espace des configurations de l'humain C_h :

$$C = C_r \times C_h$$

L'espace des configurations C contient toutes les configurations autorisées par la cinématique du robot et de l'humain. Ainsi la résolution du problème de transfert implique de trouver une configuration $q_{trans} = (q_r, q_h) \in C$. Les configurations q_r et q_h spécifient les paramètres de manipulation et de navigation du robot et de l'humain. La configuration q_{trans} appartient à un sous-espace des configurations faisables $C_{feas} \in C$. Ce sous-espace est une restriction de C concernant les contraintes énumérées ci-dessous :

Sans collision : le robot et l'humain doivent être sans collisions à la configuration de transfert q_{trans}. Les auto-collisions, collisions avec les obstacles et avec l'humain doivent être évitées. Ce sous-espace est nommé C_{libre}.

Saisie de l'objet : l'objet échangé doit être saisi par les deux partenaires à q_{trans}, c'est à dire que la pince du robot et la main de l'homme doivent saisir l'objet. Ce sous-espace est nommé C_{saisie}.

Stabilité : le robot et l'humain doivent être stables à q_{trans}, en ce qui concerne la loi de Newton de la mécanique. Ce sous-espace est nommé c_{stab}.

Accessibilité : cette contrainte correspond à l'existence d'un chemin sans collision entre la configuration de transfert q_{trans} et la configuration initiale $q_{init} = (q_r^{init}, q_h^{init})$. L'ensemble des configurations accessibles à partir de q_{init} est défini par le sous-espace nommé C_{access}

L'ensemble de toutes les configurations d'échange faisables est l'intersection des quatre sous-espaces :

$$C_{feas} = C_{libre} \cap C_{saisie} \cap C_{stab} \cap C_{access}.$$

Les différentes contraintes ne restreignent l'espace des configurations qu'à des mouvements de transfert d'objet faisables. Dans ce sous-espace de C de nombreuses configurations sont indésirables parce qu'elles ne respectent pas les protocoles sociaux ou ne tiennent pas compte des préférences de l'humain.

Dans la section suivante, nous détaillons quelques propriétés importantes qui doivent être prises en compte afin de générer des configurations valides et intuitives qui respectent les règles sociales et la théorie plus formelle de la *proxémique*.

4.1.3 Contraintes d'interaction homme-robot

Afin de tenir compte de la sécurité de l'interaction et de la lisibilité des intentions du robot, Sisbot et al ont introduit dans [Sisbot 07a] un ensemble de contraintes HRI qui s'appuient sur les notions de la théorie *proxémique* [Hall 63] ainsi que sur des études d'utilisateurs telles que [Koay 07]. Dans le chapitre précédent, nous avons utilisé ces contraintes pour générer des mouvements de manipulation en interaction proche avec l'humain. Ces contraintes sont intégrées pour produire des configurations de transfert q_{trans} sûres et confortables. Par ailleurs, nous tenons également compte d'un deuxième groupe de contraintes qui sont liées aux mouvements de l'homme et du robot pour y accéder et permettent d'évaluer l'effort et le gain en terme de rapidité de transfert. Ces contraintes peuvent également être intégrées à l'algorithme grâce à des fonctions de coût.

Confort de la configuration d'échange

Tout d'abord, nous considérons les contraintes qui pénalisent la génération de configurations inconfortables. Ces critères tels que la proximité du robot [Hall 63], la visibilité du robot [Koay 07], le confort musculo-squelettique de l'homme [Marler 05], ont été combinés dans le calcul du point de transfert pour un échange d'objet [Sisbot 07b].

Confort contre fluidité de l'échange

Un mouvement complet d'échange d'objet comprend la navigation du robot et de l'homme pour atteindre la configuration d'échange q_{trans}, représentée par les chemins τ_r et τ_h. Nous considérons donc des contraintes sur ces mouvements pour générer un effort raisonnable de l'humain et un échange fluide.

Les contraintes sur l'effort de l'humain dépendent de deux paramètres qui sont la longueur du chemin τ_h que l'humain aura à parcourir et le besoin de l'homme de se lever. Plus le chemin est long, plus l'effort est supposé important. Nous nommons le coût correspondant c_{mot}. Les contraintes de fluidité pénalisent les échanges trop longs en favorisant des plans efficaces. Nous considérons un coût c_{temps} lié à la valeur maximale du temps pris par le robot ou l'humain pour atteindre la configuration q_{trans}.

Paramètre de mobilité

Certaines de ces propriétés, telles que le déplacement de l'homme et la durée de l'échange peuvent être en contradiction les unes avec les autres. Pour équilibrer l'impact des différentes propriétés sur la qualité du plan, nous introduisons un paramètre de mobilité reflétant les capacités physiques du receveur humain et son empressement d'obtenir l'objet. En effet la durée de transfert peut générer une gêne si elle ne correspond pas aux préférences de l'humain : *empressement* ou *urgence* d'obtenir l'objet. La valeur de la mobilité équilibre les contraintes de mouvement et de confort pour favoriser les plans plus rapides, ce qui entraîne le coût final défini comme suit :

$$c = (c_{mot} + c_{conf}) * (1 - m) + c_{temps} * m$$

où $m \in [0\ 1]$ est le facteur de mobilité. Les contraintes d'interaction et leurs fonctions de coûts correspondantes sont évaluées au cours du processus de planification et sont combinées selon les préférences de l'homme. Elles sont modélisées par le paramètre de *mobilité*.

4.2 Algorithme

Cette section présente le planificateur de transfert qui a été développé pour calculer des configurations de transfert d'objet homme-robot faisables tout en tenant compte des contraintes d'interaction présentées ci-dessus. L'approche repose sur une combinaison d'algorithmes de grilles et d'échantillonnage qui raisonnent sur les modèles géométriques de l'humain et du robot et des obstacles dans l'espace de travail. Après un pré-traitement qui calcule des grilles dans l'espace de travail, la méthode consiste à échantillonner des configurations q_{trans}, à évaluer leur coûts en regard des contraintes HRI et à retourner le plan évalué de coût minimal.

Les principales étapes du planificateur de transfert sont esquissées dans l'algorithme 4.1. Une phase d'initialisation, appelée `initGrids`, calcule l'accessibilité de l'humain et du robot dans le plan sous forme de deux grilles planes. Ces grilles contiennent également la distance de navigation à la position initiale pour le robot et l'homme (voir la figure 4.2). Dans cette phase, deux ensembles de configurations de transfert q_{RH}^{stand} et q_{RH}^{sit} pré-sélectionnées sont également pré-calculées pour une position debout et assise de l'humain (voir la Figure 4.3). Après la phase d'initialisation, chaque itération consiste à calculer une configuration de transfert q_{trans} qui spécifie les degrés de liberté du robot et de l'homme. Les paramètres de navigation p $= (x, y, \theta)$ de

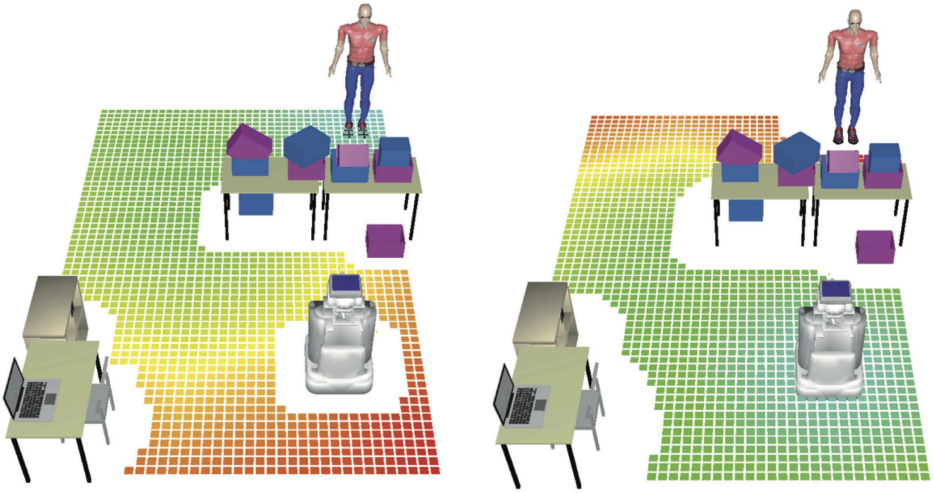
Algorithm 4.1: Calcul de plan de transfert d'objet

```
input  : Human initial position : p_h
         Robot initial position : p_r
         Mobility of the human : m
output : The human and robot handover conf : q_hand
         Human path : τ_h
         Robot path : τ_r
begin
    cost^best ← ∞
    G ← initGrids(p_h, p_r, m);
    while StopCondition() do
        p ← SampleHumanPos()
        τ_h ← DescendOnHumanGrid(p)
        q_hand ← BestFeasibleConf(p)
        if q_hand == NULL then
            continue
        p_rob ← GetRobotPos(q_hand)
        τ_r ← DescendOnRobotGrid(p_rob)
        if not τ_r == NULL then
            continue
        cost ← ComputeCost(m,τ_h,τ_r,q_hand)
        if cost > cost^best then
            continue
        else
            cost^best ← cost
            StoreBest(q_hand,τ_h,τ_r)
    return
end
```

l'être humain à q_{trans}, sont d'abord générés dans la fonction `SampleHumanPos` qui échantillonne aléatoirement une position dans l'espace accessible de l'humain. Le chemin de navigation de l'humain τ_h pour atteindre cette position est calculé par simple descente du gradient de distance dans la grille de l'humain dans la fonction `DescendOnHumanGrid`. La chemin du robot τ_r est calculé de façon similaire par une descente du gradient de distance dans la grille du robot. La position p est ensuite transformée en une configuration de transfert entièrement spécifiée q_{trans} dans la fonction `BestFeasibleConf` qui parcourt l'ensemble des configurations pré-sélectionnées chargées dans la phase d'initialisation afin de trouver une configuration valide. Ensuite, si aucune trajectoire du robot τ_r n'est trouvée entre la position initiale du robot et la configuration de transfert, l'algorithme retourne à la première étape. Sinon, le coût du plan de la solution est évalué et stocké si il améliore la solution de coût minimal calculée jusqu'ici.

Le procédé boucle sur chaque étape jusqu'à ce que le critère d'arrêt soit satisfait. Dans l'implémentation actuelle les conditions d'arrêt combinent deux critères qui sont le temps maximum ou l'amélioration minimale de la meilleure solution.

Dans les prochains paragraphes, nous donnons plus de détails sur le traitement effectué au cours de la phase d'initialisation et sur les trois étapes de l'algorithme. Nous décrivons aussi



(a) Grille de distance de l'homme (b) Grille de distance du robot

FIGURE 4.2 – La propagation de distance (a) est centrée sur l'humain (b) est centrée sur le robot. les cellules vertes correspondent aux cellules proches et les cellules rouges aux cellules lointaines.

certains pré-traitements supplémentaires qui peuvent être faits pour accélérer l'échantillonnage des solutions.

4.2.1 La propagation de distance et l'initialisation

L'algorithme comprend une phase de pré-traitement de l'espace de travail effectuée dans la fonction `initGrids`. En effet, afin de faciliter l'évaluation des contraintes de faisabilité et d'interaction avec un moindre coût de calcul, deux grilles planes sont construites. Ces grilles, représentées sur la Figure 4.2, pour le robot et pour l'humain, fournissent une approximation de l'espace libre et la distance de navigation à la position initiale. Ceci permet de déterminer sans calcul l'accessibilité d'une configuration de l'humain ou du robot et sa distance de navigation à la position initiale pendant le processus d'échantillonnage.

Les grilles sont calculées à partir des cylindres englobants du robot et de l'homme dans des postures de repos. Ces postures, illustrées sur la Figure 4.2 correspondent aux configurations de navigation des bras. Pour le robot, les bras sont repliés sur la base afin de minimiser le risque de collision. Pour l'homme, les bras sont droits le long du corps proches d'une posture de marche. Une cellule est marquée comme libre si le cylindre englobant du robot ou de l'humain placé en son centre ne chevauche pas les obstacles de l'espace de travail. L'espace accessible de la position initiale est calculé par une technique de propagation de distance standard [Latombe 91b]. Cette méthode calcule également la distance de navigation à la configuration initiale pour chaque cellule. La Figure 4.2 représente la distance propagée sur la grille du robot et de l'humain à partir de leurs positions initiales avec un code couleur. Les cellules vertes sont à proximité de la position initiale et les cellules rouges sont plus lointaines.

La phase d'initialisation charge également un ensemble de configurations de transfert prédéfinies

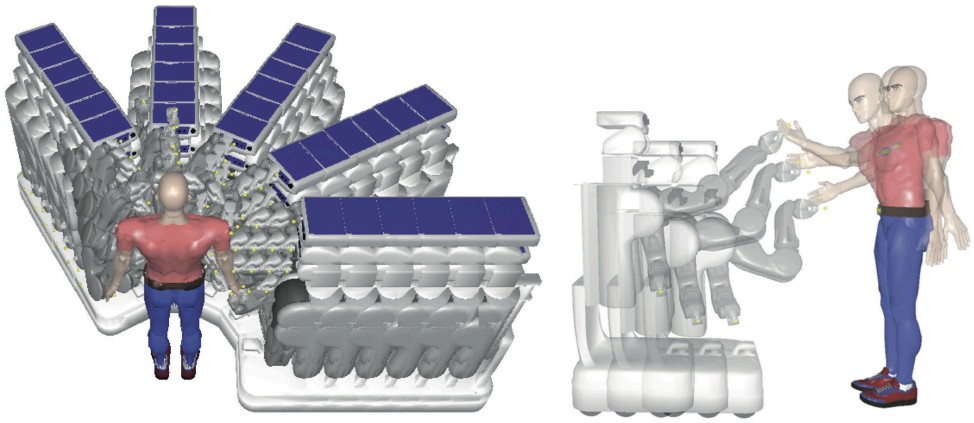


FIGURE 4.3 – Les configurations pré-sélectionnées du robot sont relatives à l'humain en position debout (des configurations similaires sont générées pour l'humain assis).

illustrées sur la Figure 4.3. Ces configurations d'échange robot-homme sont nommées q_{RH} dans le reste du chapitre. Elles sont choisies hors-ligne et ne dépendent pas de l'espace de travail, ni de la position absolue de l'humain et du robot. Chaque configuration est définie relativement à la position de l'homme et se compose des degrés de liberté de l'humain et du bras du robot.

4.2.2 Echantillonage des positions de l'humain

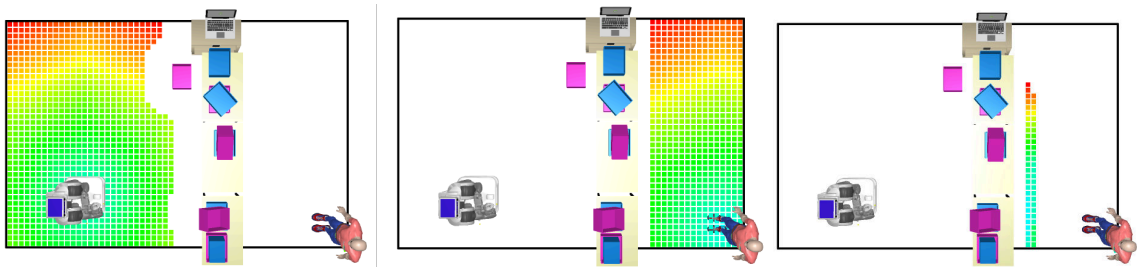
La première étape de chaque itération est l'échantillonnage d'une position et orientation de l'humain $p = (x, y, \theta)$ dans l'espace accessible stocké dans la grille pré-traitée. Afin de sélectionner ce triplet, dans la fonction `sampleHumanPos`, une cellule est choisie, puis un point est échantillonné à l'intérieur de la cellule et une orientation est choisie aléatoirement. Étant donné que chaque position p conduit à une seule configuration de transfert, il est important de sélectionner les positions qui donnent de meilleures solutions. Nous présentons en section 4.2.4 deux améliorations de la phase de pré-traitement pour la sélection de la cellule, et de l'orientation de l'humain.

4.2.3 La meilleure configuration faisable

Les configurations q_{RH} illustrées sur la Figure 4.3 sont triées en fonction du coût c_{conf} (voir section 4.1.3). Pour une position de l'homme p, l'ensemble des configurations q_{RH} est parcouru dans la fonction `BestFeasibleConf` en traitant d'abord les configurations avec un faible coût c_{conf}, la première configuration faisable est sélectionnée (c'est à dire sans collision et accessible du robot). Ce processus permet de trouver des configurations d'échange dans des espaces contraints par exemple un passage étroit dans un mur qui sépare deux parties déconnectées de l'espace de travail.

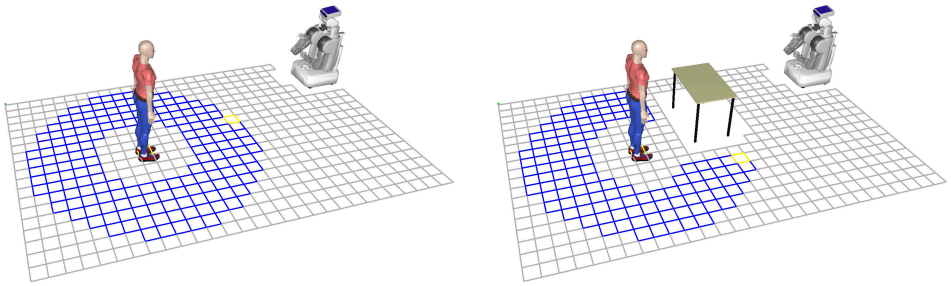
4.2.4 Réduire l'espace de recherche et biaiser l'échantillonnage

Quand l'humain et le robot se trouvent dans des parties séparées de l'espace de travail, comme illustré sur l'exemple de la Figure 4.4, l'algorithme échantillonne les cellules de la grilles accessible



(a) Espace accessible du robot (b) Espace accessible de l'humain (c) Grille fusionée

FIGURE 4.4 – L'humain et le robot se trouvent dans deux parties séparées par des tables de l'espace de travail. L'espace accessible du robot (a) et de l'humain (b) et des grilles fusionnées (c) sont calculés.



(a) Region courronne (b) Solution sur le coté

FIGURE 4.5 – Les positions de transfert d'objet faisable (en bleu) et la position faisable la plus proche du robot $cell_{min}$ (en jaune) pour une position donnée en tenant compte des capacités de l'humain et du robot.

par l'humain sans tenir compte de l'accessibilité du robot et de ses capacités de manipulation. La direction de l'homme est également échantillonnée de manière aléatoire sans tenir compte de la direction d'approche du robot.

Afin d'accélérer et d'améliorer la génération de configurations de transfert faisables, nous proposons deux améliorations à la version de base. Tout d'abord, nous construisons une grille fusionnée à partir des cellules accessibles de l'homme. Les cellules ne pouvant pas conduire à une configuration de transfert valide sont supprimées. Ensuite, une valeur qui se rapproche du coût total de la solution de transfert, détaillée section 4.2.4.1, est pré-calculée et stockée dans la grille. L'échantillonnage de la position de l'humain est ensuite effectué dans cette grille fusionnée et biaisée vers les régions à faible coût de l'espace accessible. L'échantillonnage de la direction de l'humain est également biaisée vers une estimation de la direction d'approche du robot également détaillée dans la section 4.2.4.1.

Grille Fusionnée

Pour générer la grille fusionnée de l'espace d'accessibilité de l'homme et du robot, les distances minimales et maximales (min,max) entre l'homme et le robot dans l'ensemble de configuration

pré-sélectionnées q_{RH} Figure 4.3 sont déterminées. Pour chaque configuration d'échange, la distance entre l'homme et le robot est calculée entre la position 3D du centre du bassin de l'humain et du centre de la base mobile du robot projetée sur sol. Les valeurs min et max sont ensuite utilisées pour définir une région en couronne autour de chaque cellule de l'espace accessible comme illustré dans la Figure 4.5. Puis chaque cellule accessible dans la grille de l'humain dont la couronne ne chevauche pas de cellule dans la grille d'accessibilité du robot est supprimée de la grille fusionnée. De cette façon, les positions qui ne peuvent pas conduire à des configurations de transfert non valides sont supprimées. Par conséquent, la grille fusionnée résultante telle que représentée sur la Figure 4.4 traduit l'accessibilité de l'homme et intègre également des informations complémentaires concernant la faisabilité de la position candidate pour le transfert d'objet.

4.2.4.1 Biais de l'échantillonnage

Afin de biaiser l'échantillonnage de la position de l'humain vers les candidats qui produiront une meilleure configuration de transfert, chaque cellule de la grille fusionnée est associée à un coût c_B de la façon suivante :

$$c_B = c_{mot} * (1 - m) + c_{temps} * m$$

où m est le paramètre de mobilité. c_B se rapproche du coût c présenté dans la section 4.1.3 . Il est utilisé pour évaluer la qualité d'une configuration de transfert d'objet. Le coût c ne peut être précalculé car il dépend de la configuration des bras de l'homme et du robot. La meilleure configuration dépend donc de la forme 3D des obstacles présents dans l'environnement et peut s'avérer difficile à déterminer dans le cas d'un passage étroit. Dans la section 4.3, nous montrons des exemples de stratégies d'échanges qui illustrent cette difficulté.

Afin de biaiser l'échantillonnage de la direction de l'humain, la cellule valide qui minimise le mouvement du robot depuis sa position initiale jusqu'à la couronne est stockée dans la grille fusionnée (en jaune dans la Figure 4.5). Lors de l'échantillonnage, θ est déterminé de telle sorte à ce que l'humain soit dirigé vers le centre cette cellule. L'angle γ entre la position 2D de l'humain et le centre de cette cellule est calculé. Une quantité aléatoire est ajoutée avec une probabilité faible de la façon suivante :

$$\theta = \gamma + rand^n * sign * \pi$$

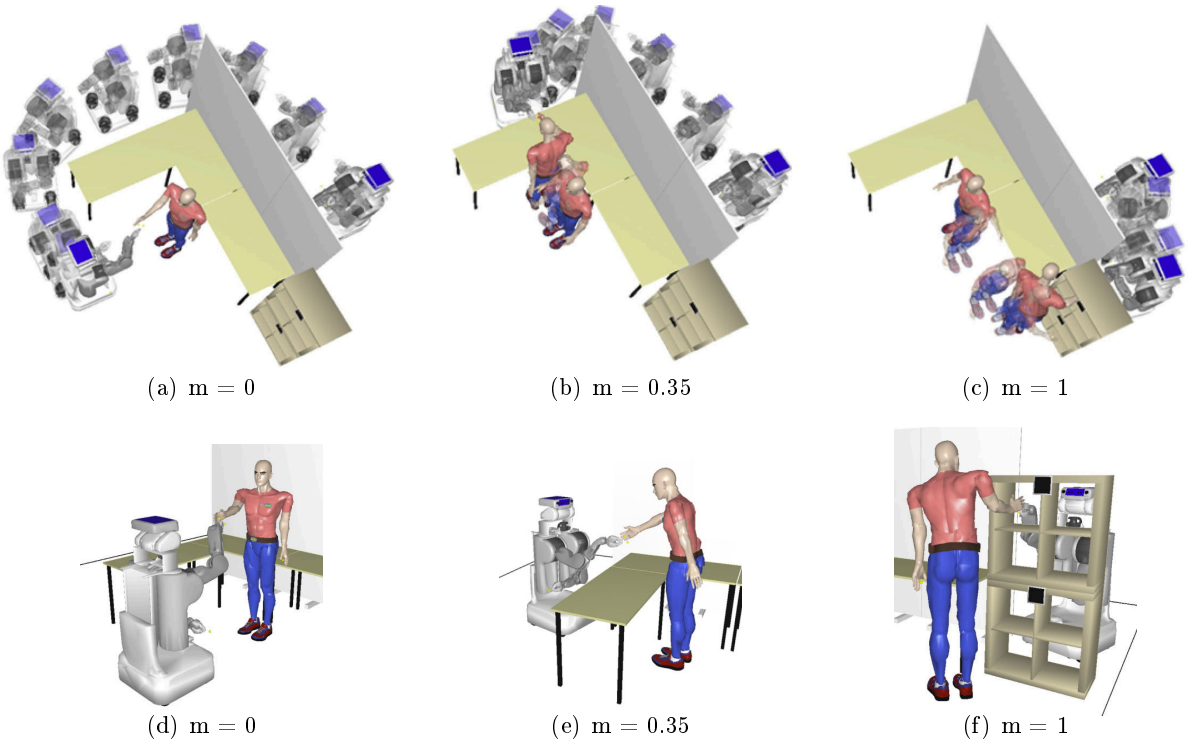
où $rand$ et un nombre aléatoire $\in$ [0 1] et $sign$ et choisi aléatoirement à -1 ou 1. n permet de contrôler l'intensité du biais. La section suivante fournit des résultats en simulation de cet algorithme avec des valeurs différentes du paramètre m. L'efficacité et les limites des améliorations de la version de base sont également analysées.

4.3 Resultats en simulations

Dans cette section, nous présentons une évaluation des capacités de l'algorithme à trouver des configurations de transfert pour le robot PR2 dans des espaces de travail contenant des meubles, des tables et des étagères. Nous rapportons les stratégies produites par le planificateur en analysant la convergence du coût de la meilleure solution produite pour différentes valeurs de m. Nous discutons également de l'utilisation des différentes variantes de pré-traitement et des questions de discrétisation.

Afin d'évaluer les performances de l'algorithme, il a été mis en œuvre ainsi que les environnements de test dans le logiciel de planification de mouvement *Move3D* [Simeon 01] et la simulation a été effectuée sur un processeur INTEL cadencé à 2,26 GHz.

4.3.1 Influence du paramètre de mobilité



(a) m = 0 (b) m = 0.35 (c) m = 1

(d) m = 0 (e) m = 0.35 (f) m = 1

FIGURE 4.6 – Trois valeurs du paramètre de mobilité sont utilisées pour générer trois stratégies de transfert d'objet. Les premières images montrent les trajectoires résultantes alors que les trois images du bas montrent les configurations finales de transfert d'objet qui prennent en compte les obstacles 3D.

La Figure 4.6 montre trois stratégies de transfert qui ont été calculées pour le même problème en utilisant trois valeurs du paramètre m. Pour de faibles valeurs de m, l'être humain est censé être moins impliqué, ce qui lui demande un effort minimal. Au contraire les valeurs élevées de m sont supposées exiger plus d'effort et de participation de l'humain résultant en des stratégies de transfert plus rapides.

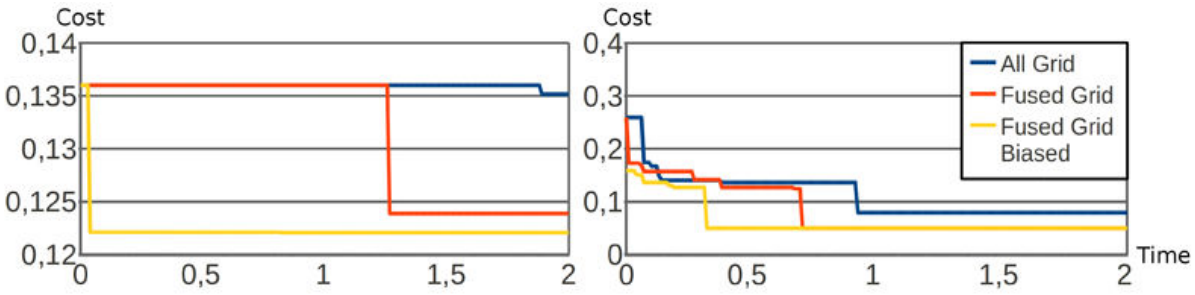


FIGURE 4.7 – Courbes de convergence avec deux réglages de m, utilisant les trois variantes de pré-processing qui correspondent à trois stratégies d'échantillonnage sur le scénario représenté sur la Figure 4.6 (m=0.35 droite, m=1 gauche).

- $m = 0$: Un long chemin est généré pour le robot et l'humain n'a pas à se déplacer
- $m = 0.35$: Un chemin plus court pour le robot jusqu'à une configuration faisable par dessus une table est rendu possible grâce à un léger déplacement de l'humain.
- $m = 1$: Un effort partagé entre le robot et l'humain autorise un transfert d'objet à travers les étagères.

Nous constatons que le planificateur est capable de trouver différentes configurations possibles pour la même position initiale de l'humain et du robot. Les plans résultants prennent en compte la faisabilité du mouvement en utilisant les modèles 3D de l'être humain et du robot, même si la planification de mouvement de navigation est effectuée dans l'espace cartésien 2D.

4.3.2 Analyse des performances des variantes de pré-traitement

Les courbes de la Figure 4.7 montrent l'amélioration du coût sur deux secondes sur une passe de l'algorithme qui correspond à un millier de positions de l'humain échantillonnées dont cinq cents stratégies de transfert entièrement évaluées. Le coût de la meilleure solution est affiché sur une durée de 2 sec. Le cas le plus simple ($m = 0$) n'est pas représenté sur la Figure 4.7 car toutes les variantes convergent vers la solution affichée au bout de quelques itérations. Toutefois, pour les deux cas les plus complexes les résultats montrent la difficulté de trouver des configurations par dessus la table ($m = 0{,}35$) ou à travers les étagères ($m = 1$).

Afin de générer un transfert plus direct et moins coûteux, la version de base exige beaucoup plus d'itérations que les autres variantes. La méthode de base échantillonne des configurations de transfert dans l'espace libre semblables à celles de la Figure 4.6.a et 4.6.d, mais placées ailleurs dans l'espace libre. Elle se concentre donc moins sur l'échantillonnage de configuration à proximité des passages étroits qui permettent de trouver des stratégies de bas coût. Grâce aux grilles fusionnées l'algorithme génère plus d'échantillons à la frontière de l'espace libre (à proximité des tables) et grâce au biais il découvre plus facilement les solutions à travers les étagères. La Figure 4.8 montre des résultats similaires sur 300 passes du planificateur qui confirment ces résultats. En moyenne une solution de bonne qualité est obtenue en moins de 1 sec avec la grille fusionnée et le biais de l'échantillonnage.

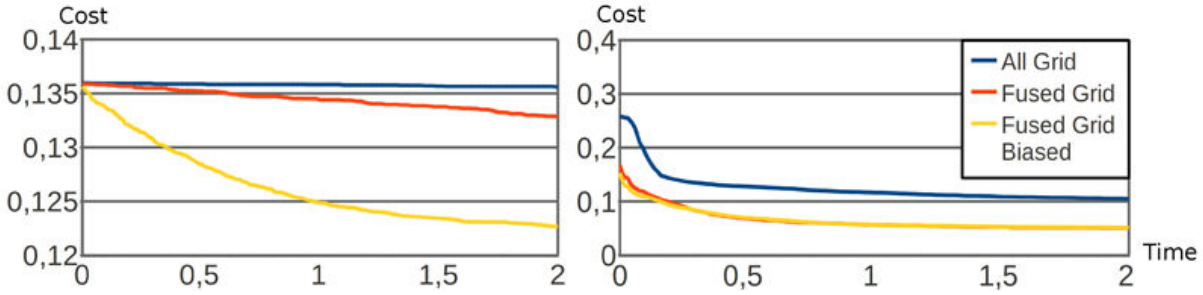


FIGURE 4.8 – Convergence moyennes sur 300 éssais, avec les mêmes réglages que ceux de la Figure 4.7 (m=0.35 right, m=1 left)

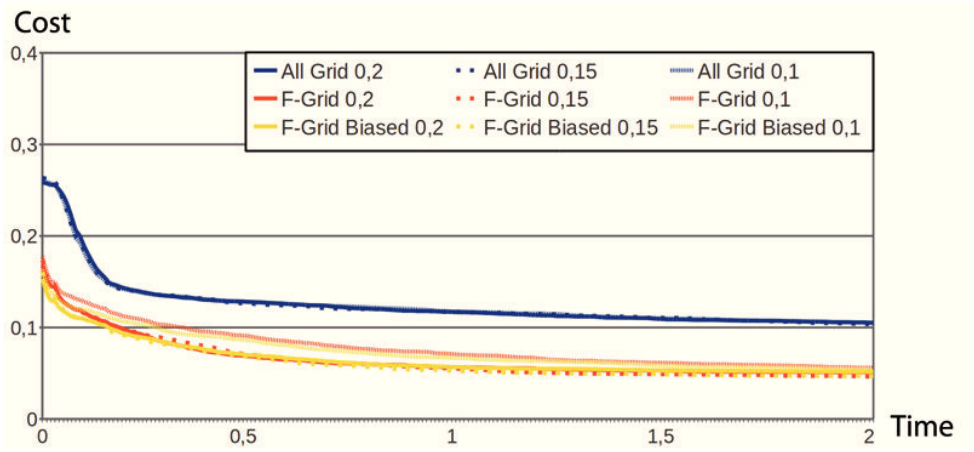


FIGURE 4.9 – Moyenne de convergence sur 300 essais sur le scénario de la Figure 4.6 avec m=1. Trois différentes tailles de grille, 10, 15, et 20 cm sont superposées.

4.3.3 Influence de la discretisation

La Figure 4.9 montre que pour le cas $m = 1$, la taille de la grille a peu d'influence sur la performance de la phase d'échantillonnage. L'influence est limitée à la phase de pré-traitement pour laquelle les temps de calcul de la propagation de distance et de la grille fusionnée dépendent de la résolution. Le traitement de base nécessite 200 msec (resp. 660 msec) pour une résolution de 20 cm (10 cm resp.) et le coût de calcul supplémentaire de la grille fusionnée est de 68 msec (resp. 1109 msec). Ainsi, en utilisant une résolution de 20 cm, le pré-traitement reste en dessous de 1 sec mais il peut devenir plus élevé pour 10 cm.

Les performances de l'algorithme ci-dessus tendent à indiquer que cette approche pourrait être utilisée pour adapter dynamiquement le plan de mouvement calculé afin de tenir compte d'éventuels mouvements de l'homme. Le paramètre de mobilité pourrait aussi être adapté si le mouvement de l'humain ne semble pas compatible avec la position de transfert initiale.

4.4 Validation experimentale

Nous avons conduit une expérience HRI pour évaluer la pertinence de l'approche. Les participants ont été soumis à deux types de solution calculées par notre planificateur. La première correspond à un temps de réalisation de la tâche le plus court possible au prix d'un effort conséquent demandé à l'humain. La seconde correspond au plan qui minimise l'effort de l'homme au prix d'une performance globale plus faible. Des mesures objectives et subjectives sont discutées afin de valider notre l'hypothèse suivante :

Hypothèse

Nous nous attendons à ce que la prise en compte de la mobilité de l'humain se traduise par des transferts d'objets plus fluides et plus efficaces. La mobilité du récepteur humain dépend certainement de la tâche et d'autres paramètres intrinsèques comme ses capacités physiques ou son implication dans une autre tâche. C'est pourquoi nous faisons l'hypothèse qu'un réglage approprié du paramètre de mobilité se traduise par une interaction plus douce et plus confortable.

4.4.1 Principe de l'expérience

L'expérience comprend 34 sujets recrutés au LAAS-CNRS à Toulouse, d'un âge moyen de 28 ans. Dans le groupe 82% étaient de sexe masculin et 18% étaient de sexe féminin. Les sujets ont été confrontés à un seul transfert d'objet. À leur arrivée, les sujets ont été informés du principe de l'expérience sans dévoiler l'hypothèse pour ne pas introduire de bais. Leur interaction avec le robot lors de la réalisation de la tâche a ensuite été enregistrée en vidéo. Un questionnaire leur été remis après l'échange d'objet.

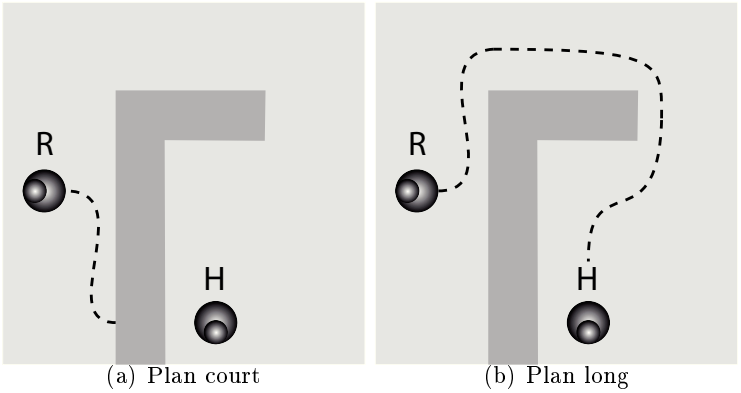
Platforme

Nous avons utilisé un robot PR2. Ce robot est équipé de deux bras manipulateurs à 7 degrés de liberté, d'une base mobile omnidirectionnelle et d'un télémètre à balayage laser pour la localisation et la détection d'obstacles. L'humain est détecté en utilisant un Kinect de Microsoft. Nous utilisons la navigation de ROS [Quigley 09] disponible pour le PR2, elle s'appuie à la fois sur une carte de l'environnement et sur la détection d'obstacles par laser.

Notre algorithme de planification est mis en œuvre dans *Move3D*. Il est utilisé pour planifier la configuration de transfert et les mouvements du bras. Les trajectoires du bras sont exécutées avec un contrôleur SoftMotion [Broquère 08] qui limite le mouvement en jerk, accélération et vitesse. Les objets mobiles sont localisés avec un module de détection d'étiquettes basé sur ARToolKit [Prince 02]. Nous contrôlons la tête du robot et diffusons du son sur les haut-parleurs du robot.

Scenarios

Les deux types de transfert d'objet sont représentés schématiquement sur la Figure 4.10. Dans les deux cas, les participants sont assis sur une chaise derrière un mur qui présente une fente par



(a) Plan court (b) Plan long

FIGURE 4.10 – Nous avons étudié deux cas de transfert d'objet. Dans le premier cas, le robot propose l'objet à travers le mur. Le mouvement est plus court mais nécessite que l'humain se lève pour attraper l'objet. Dans le second cas, le robot vient plus proche de l'humain en contournant des obstacles produisant un large détour.

laquelle doit passer l'objet à échanger. Le robot commence par ramasser l'objet représenté par une petite balle en plastique. Dans le cas d'une valeur élevée de la *mobilité* (chemin court), le robot utilise l'accès direct à l'humain à travers le mur pour transmette l'objet. L'humain sur la chaise n'est pas en mesure d'atteindre l'objet et doit donc se lever et marcher pour saisir l'objet. Dans le second cas, planifié avec une valeur faible de la *mobilité* (chemin long), le robot effectue un mouvement de détour évitant l'obstacle en forme de L et l'humain peut recevoir l'objet en restant assis.

Tâches

Nous demandons dans un premier temps aux participants de s'asseoir sur une chaise et nous leur expliquons que le robot va leur remettre une balle qu'ils doivent placer à l'intérieur d'un tube en plastique placé à côté de la chaise. Une des deux tâches supplémentaires suivantes est attribuée aux participants : La première tâche est censée inciter les sujets à réaliser rapidement le transfert de la balle et la deuxième tâche incite le sujet à accorder moins d'attention au transfert rapide de l'objet.

La première tâche, qui correspond à une mobilité haute, consiste à regarder un chronomètre et à essayer de réaliser le transfert le plus rapidement possible. La deuxième tâche, qui correspond à une mobilité faible, consiste à jouer à un jeu de sudoku. Les participants se voient donc attribuer l'un des 4 types de scénarios mentionnés ci-dessus. Dans le reste de cette section, les quatre cas sont dénommés par les lettres suivantes :

A : Stratégie courte avec chronomètre
B : Stratégie courte avec 'sudoku'
C : Stratégie longue avec chronomètre
D : Stratégie longue avec 'sudoku'

4.4.2 Evaluation

Afin d'évaluer la fluidité et l'efficacité de l'interaction, deux types de données ont été extraites des enregistrements vidéos. Nous avons évalué le temps de réaction en détectant l'instant auquel les sujets commencent à se déplacer pour prendre l'objet. Nous avons ensuite évalué le temps total nécessaire pour effectuer la tâche. Nous avons pris en compte deux événements, le début de la navigation du robot et l'instant où la balle pénètre dans le tube. La qualité de l'interaction est évaluée par un ensemble de critères subjectifs recueillis en compilant les réponses des questionnaires.

Questionnaire

Les questionnaires remis aux participants comportent un ensemble de données générales (âge, sexe, humeur générale) suivi d'un ensemble de questions de trois types : ouvertes, fermées et d'évaluation. Huit questions à choix multiples ont été posées dont cinq permettant aux participants d'évaluer l'un des critères par une réponse graduée. Les questions d'évaluation concernent le confort. e.g. si la distance du robot lors de l'échange semble appropriée et si le robot les a surpris, voir effrayé. Afin d'évaluer la mobilité des participants nous leur avons demandé si ils avaient ressenti de l'empressement pour obtenir l'objet et si l'attente pendant la navigation du robot leur avait semblé courte ou longue. Des questions avec des réponses fermées (oui ou non) leurs ont été posées pour déterminer si ils avaient bien compris le lieu où l'objet allait être échangé et si ils avait trouvé le lieu naturel. Enfin, les participants devaient indiquer s'ils auraient préféré l'une des solutions de remplacement de la Figure 4.10 fournie pendant l'enquête.

4.4.3 Résultats

Dans cette section, nous comparons les temps mesurés et les réponses de l'enquête pour étudier la validité de notre hypothèse. Les résultats sont présentés sur les Figures 4.13, 4.14 et 4.15.

Temps et attente

La Figure 4.13 présente la durée totale de la tâche et indique que le transfert d'objet a été réalisé de manière plus rapide avec des mouvements de type A et B. Les temps de réaction indiqués dans le graphique 4.15 sont plus courts pour les participants auxquels il était donné une contrainte de temps. Par contre, les temps sont plus longs quand les participants doivent remplir un 'sudoku'. Ceci suggère que les sujets étaient plus conscients et plus prompts à accomplir la tâche avec une contrainte de temps donnée. Nous pensons que cette observation particulière sur le comportement des sujets corrobore la première partie de notre hypothèse qui postule que la tâche et le contexte module la mobilité du récepteur humain.

L'attente ressentie lors de la phase de navigation, rapportée dans la Figure 4.13, est corrélée à la durée totale du transfert d'objet. Cependant, la durée de l'interaction des participants de type A et B ne diffère que par le temps de réaction. Une explication possible est que le temps

(a) Début (b) Signal

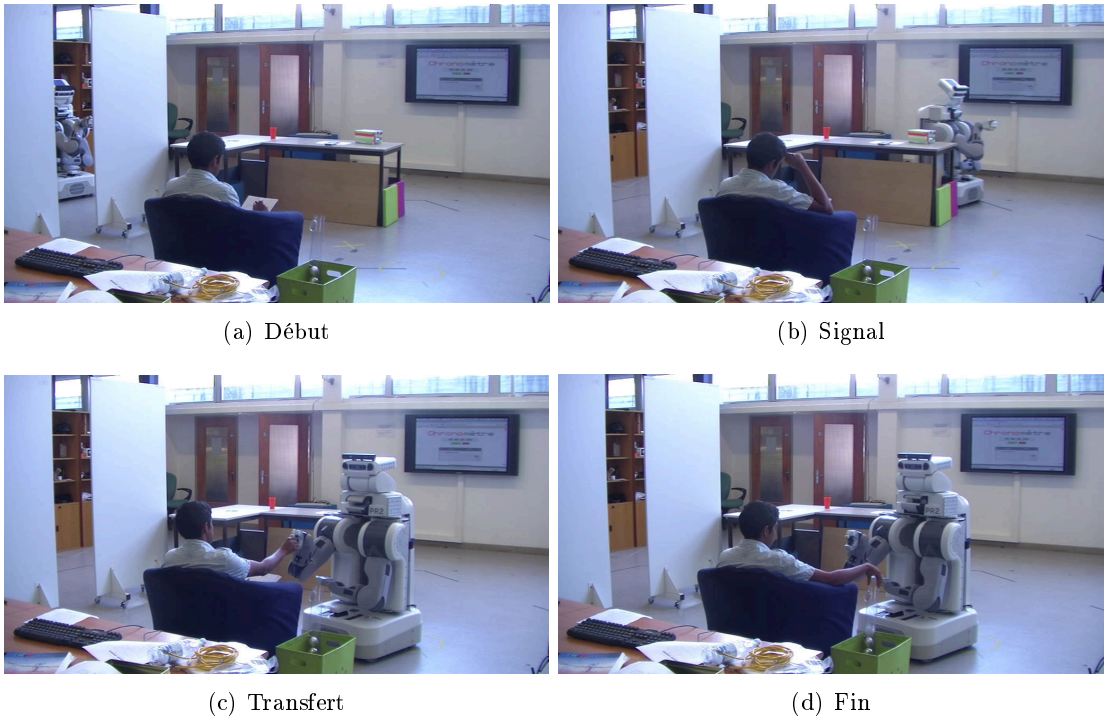
(c) Transfert (d) Fin

FIGURE 4.11 – Type de scénario B : le robot transmet un objet à travers le mur

d'attente ressenti par les participants de type B est causé par l'inattention qui résulte en un temps de réaction plus élevé et donc une attente plus longue. Par ailleurs, les vidéos montrent que les temps d'attente plus longs sont dûs à des hésitations des sujets qui ne comprennent pas qu'il doivent se lever. Il apparait donc que ne pas tenir compte de la mobilité appropriée dans le transfert d'objet peut réduire en partie l'efficacité de la tâche en raison de la perte de lisibilité du comportement des robots.

Lieu du transfert d'objet

Nous n'avons pas informé les sujets que le robot pouvait leur demander de contribuer au transfert d'objet en se levant de leur chaise. Une partie de l'expérience était d'observer comment réagissent les humains à une telle décision et comment ils l'interprètent. Nous leur avons demandé d'évaluer si ils avaient bien compris le lieu de l'échange. Les emplacements pour la stratégie courte sont également bien compris comme indiqué dans la Figure 4.13 bien que perçus comme moyennement naturels. Pour les cas de types C et D, la compréhension de l'emplacement d'échange n'est pas perçue de la même manière en fonction de la tâche. Ceci montre que l'in-



(a) Début (b) Signal

(c) Transfert (d) Fin

FIGURE 4.12 – Type de scénario D : le robot transmet l'objet avec un large détour

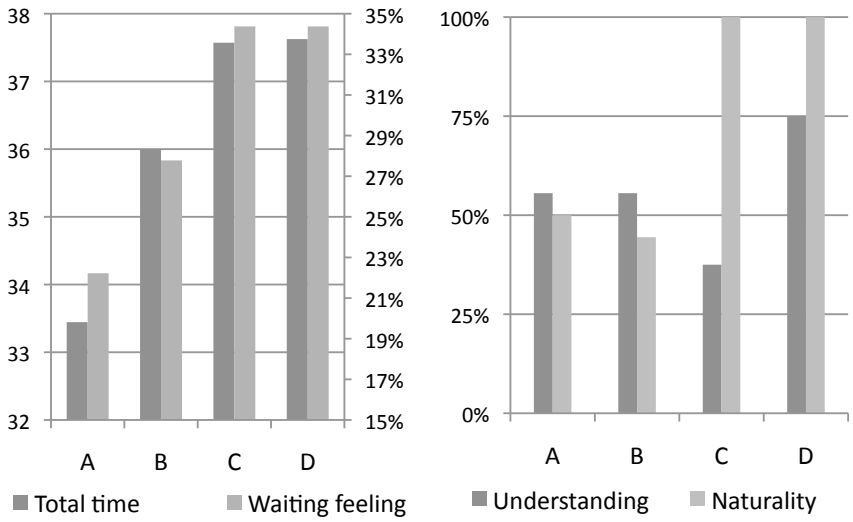
tuition qu'un individu peut avoir sur le lieu d'échange de l'objet change en fonction de la tâche qu'il effectue.

Bien que le mouvement avec un large détour soit perçu comme très naturel, les participants ayant une contrainte de temps s'attendent à une collaboration avec le robot et se tournent régulièrement vers le robot pendant son déplacement. Le choix effectué par le robot était aussi difficile à comprendre pour les participants de type C, en partie parce que le détour pris par le robot n'est pas approprié à la contrainte qui consiste à effectuer l'échange rapidement.

Les sujets trouvent généralement le lieu d'échange du chemin le plus court pas explicitement naturel. Les participants confrontés au transfert d'objet plus long trouvent avec une très large majorité que l'emplacement du transfert d'objet est naturel. Cela est dû au comportement inattendu du robot. En effet, le choix de remettre l'objet dans la fente n'est pas évident et peut surprendre. Dans les questions ouvertes, un participant a déclaré qu'il a trouvé le robot "intelligent".

Confort de l'interaction

Les chemins plus courts sont perçus comme moins confortables et les distances de placement du robot pour l'échange sont perçues comme trop longues pour la plupart des participants, cela est signalé dans la Figure 4.14. Par ailleurs les sujets sont moins effrayés par le robot dans ce cas. Ceci est dû à la distance au robot qui est plus grande dans les mouvements directs, car le robot s'arrête derrière le mur. La distance d'interaction est généralement perçue comme bonne pour



(a) Time and wait (b) Hand-over location

FIGURE 4.13 – La sensation d'attente est corrélée au temps total de l'interaction (sec) alors que les mouvements avec moins d'effort sont perçus comme plus naturels que les mouvements courts.

les sujets de type C qui sont attentifs à la tâche. Les participants de type D sont généralement plus effrayés par le robot car ils sont moins attentifs à ses mouvements. Ceci est soutenu par la théorie de la *proxemique*. Le robot d'un volume important pénètre l'espace personnel du sujet qui est théoriquement plus grand lorsqu'il se concentre sur une tâche intellectuelle telle que le sudoku. Cette situation peut donc occasionner une sensation de peur.

L'hypothèse impliquerait que les sujets de type C (empressement et détour) ressentiraient moins de confort que les participants de type D (sudoku et détour) en raison de l'inadéquation entre leur empressement et le mouvement assez lent du robot. Ce n'est pas le cas parce que les sujets de type D ont été plus impressionnés, voire surpris par le robot que les participants de type C du fait de leur concentration. Il peut donc être important en plus de considérer la mobilité associée à la tâche que le robot signale sa présence quand il se trouve à proximité de l'humain afin d'obtenir une interaction plus confortable.

Empressement et préférence sur la mobilité

L'empressement et les préférences de stratégie (i.e. chemin court ou long) sont présentés dans la figure 4.15. Les réponses évaluant l'empressement montre une variance faible. La contrainte du chronomètre fixée pour la tâche supplémentaire n'était pas assez forte pour influer sur l'empressement ressentit par les sujets de l'étude. La Figure 4.15 indique le nombre de participants qui auraient préféré l'autre type de transfert. Les participants de type D sont satisfait à 100% du type de transfert qui leur à été présenté. Ceux de type B, qui montrent des temps de réaction plus élevés et donc une hésitation par rapport à la stratégie choisie par le robot montrent un intérêt très élevé pour l'autre trajet (85 % des sujets). Ceci est en accord avec l'hypothèse qui prédit qu'un réglage du paramètre de mobilité en adéquation avec les contraintes liées à la

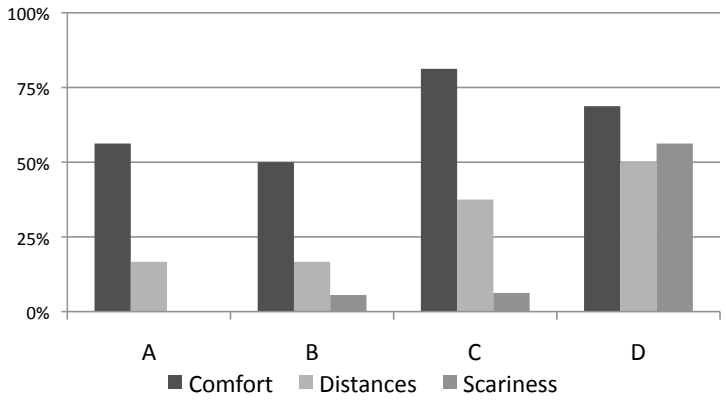


FIGURE 4.14 – Confort, distance et le lieu d'échange des objets avec la peur suscitée par l'interaction

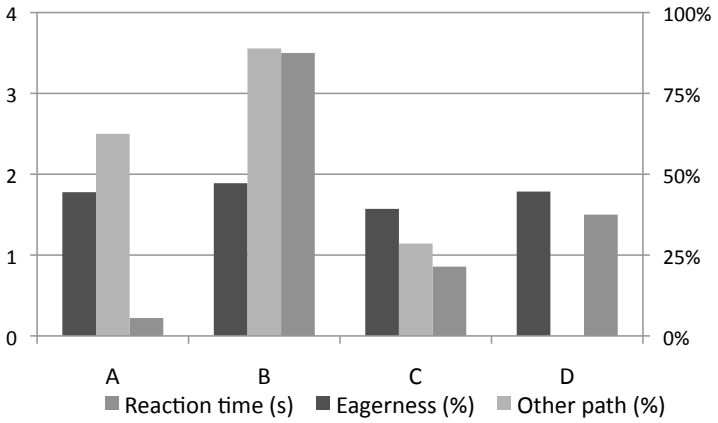


FIGURE 4.15 – Empressement de prendre l'objet, temps de réaction et choix d'une autre stratégie

tâche est souhaitable. Cependant, plus de 50% des participants de type A et moins de 25% des participants de type C ont préféré l'autre transfert. Ceci indique également que la contrainte de temps était trop faible pour provoquer une envie forte d'avoir l'objet le plus vite possible.

4.5 Conclusions

Ces travaux ont été motivés par la nécessité de planifier les mouvements du robot pour le transfert d'objets en tenant compte des capacités de déplacement du partenaire humain. Afin de prendre en compte ses préférences, nous avons introduit une nouvelle propriété que nous avons appelé mobilité qui qualifie la volonté de l'humain de participer à l'effort lors de l'échange. Pour traiter ce type d'échange d'objet, nous avons introduit un nouveau problème de planification de mouvement que nous appelons le problème du partage d'effort dans le transfert d'objet homme-robot. Nous avons formalisé ce problème et proposé un algorithme permettant de le traiter. La nouveauté de cette approche consiste à planifier les mouvements de l'humain. Cet algorithme est

basé sur une combinaison de techniques simples et efficaces basées sur l'échantillonnage et sur des grilles offrant une solution rapide en temps de calcul.

La pertinence de ce paramètre a été évaluée par une expérience sur des sujets humains où les participants ont été confrontés à quatre cas distincts. L'environnement a été construit de façon à mettre le planificateur dans une situation difficile. Dans un cas la tâche est planifiée pour être plus rapide en contre partie d'un effort fourni par le sujet pour effectuer la tâche. Le robot construit un plan partagé où il remet l'objet à travers un passage étroit trop petit pour que lui-même ou l'humain puisse le traverser. Dans l'autre cas le confort de l'humain est privilégié, le robot contourne les obstacles et vient donner l'objet à l'humain sans qu'il ait à se lever.

Les résultats expérimentaux montrent que la prise en compte de la mobilité de l'humain pour le transfert d'objet améliore de manière significative l'efficacité des tâches. La mise en correspondance de la mobilité de l'humain (fonction de la tâche assignée au sujet) et de celle prise en compte par le planificateur semble augmenter le confort de l'interaction. Les participants occupés à une tâche alternative ont largement préféré la situation sans effort (mobilité réduite). Les résultats indiquent que si l'on ne tient pas compte des préférences de l'homme pour planifier ses mouvements, les réactions sont plus longues en raison du comportement moins lisible du robot. Ceci est en accord avec notre hypothèse qui stipule que le réglage du paramètre de mobilité permet de meilleurs transferts d'objet robot-humain. Cependant, nous pensons que le niveau de mobilité change de manière significative avec le type de contraintes qui agit sur ce paramètre. La contrainte de chronomètre n'était pas assez forte pour réellement montrer le potentiel de l'approche.

Ces travaux ont été réalisés en collaboration avec Mamoun Gharbi. Ils ont permis d'intégrer l'architecture logicielle LAAS sur le robot PR2.

5

Architecture logicielle pour la manipulation réactive et interactive

Le projet européen DEXMART que nous avons présenté dans l'introduction nous a permis de développer et d'évaluer une architecture dédiée à la manipulation interactive. En effet, le partage de l'environnement par l'homme et le robot d'une part et la capacité du robot à collaborer d'autre part sont indispensables pour réaliser des tâches de manipulation élémentaires comme approcher un objet ou échanger un objet avec l'homme. De plus ces données impliquent la nécessité d'une approche globale du problème.

Dans ce chapitre, nous présentons d'abord l'architecture complète mise en œuvre sur le robot Jido du LAAS-CNRS. Elle incorpore différents éléments afin de saisir, déplacer et éventuellement échanger des objets avec l'humain. L'architecture intègre les algorithmes et méthodes introduits dans le chapitre 3 pour planifier des trajectoires sûres et lisibles pour l'homme mais également nos travaux pour la replanification de chemin et leur exécution en ligne. Elle permet au robot de traiter des problèmes de manipulation dans des cas contraints de manière autonome mais également de traiter des problèmes d'interactions tels que donner et recevoir des objets. L'architecture est basée sur deux composants principaux. Le premier génère des trajectoires qui prennent en compte l'homme de manière réactive. Le second déploie un système de supervision basé sur une approche comportementale introduite par notre partenaire UNINA de l'université de Naples en Italie.

L'intégration des algorithmes et techniques a été effectuée sur la plateforme présentée sur la Figure 5.1. Le robot étant capable de manipuler des objets lourds relativement rapidement, il peut s'avérer particulièrement dangereux. C'est pourquoi son architecture logicielle doit être efficace pour traiter l'interaction avec un ou plusieurs humains. L'humain étant lui même mobile,

il est important de doter le robot de la capacité de faire face à la dynamique introduite dans l'environnement par la présence des humains. Cette capacité se décline en deux composantes utiles non seulement pour traiter la tâche de la manière la plus efficace possible, mais aussi assurer la sécurité des humains. Premièrement, le robot doit surveiller le comportement de l'humain et interpréter le plus efficacement possible son intention et ses dispositions. Deuxièmement, il doit être capable de réagir par le mouvement afin d'éviter les situations dangereuses et d'augmenter le confort de l'interaction. L'homme est donc pris en compte de manière explicite à différents niveaux pour produire un comportement sûr, lisible et agréable du robot.

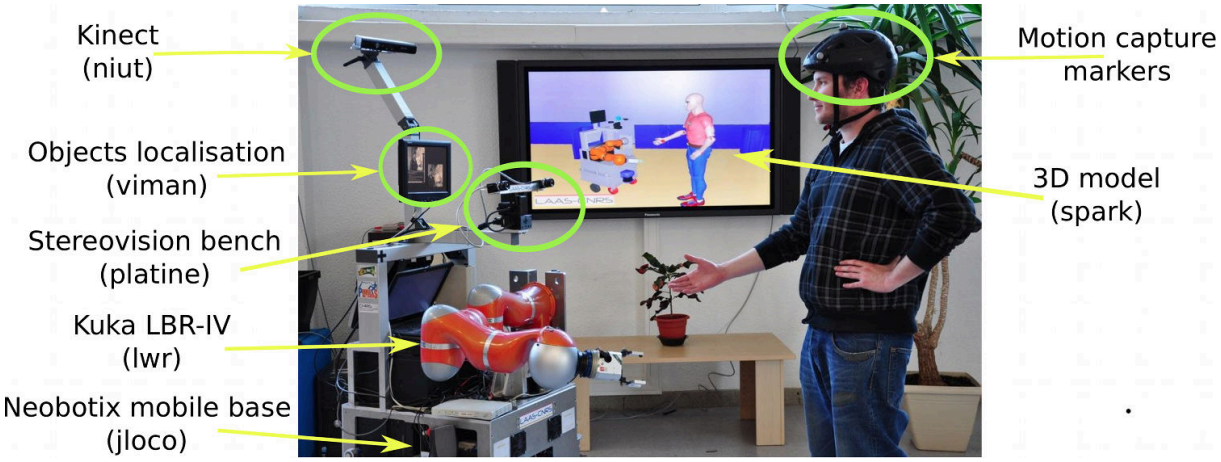


FIGURE 5.1 – Le système dédié à la manipulation intéractive à été mis en oeuvre sur la plateforme robotique Jido du LAAS-CNRS. Ce robot est équipé de différents capteurs permettant de localiser l'humain dans la scène et d'effectuer des raisonnements géométriques en temps réel.

5.1 Généralités

Pour le développement de ce système nous nous sommes principalement concentrés sur des tâches simples telles que prendre, placer, donner et recevoir un objet avec le robot Jido du LAAS qui est construit autour d'une base mobile Neobotix MP-L655 et d'un bras Kuka LWR. Il intègre une paire de caméras stéréoscopiques et un capteur de profondeur Kinect de Microsoft. Nous avons aussi utilisé un système de capture de mouvement pour identifier les mouvements de la tête des humains.

Notre approche combine plusieurs éléments de conception. Tout d'abord nous utilisons une représentation spatiale de l'interaction associée à un ensemble de cartes de coût. Les calculs sont effectués à partir de l'état cinématique et géométrique du robot et de l'homme pour évaluer les contraintes d'interaction (distance, visibilité et accessibilité). Nous avons utilisé un système attentionnel d'allocation pour la surveillance d'activité, la sélection d'action, et la régulation de l'exécution. La planification des prises, des chemins et des mouvements est réactive et adaptative et comprend des techniques de replanification. Elle est utilisée pour générer et ajuster les trajectoires de manipulation pendant l'exécution en prenant en compte les cartes de coût et l'état

attentionnel.

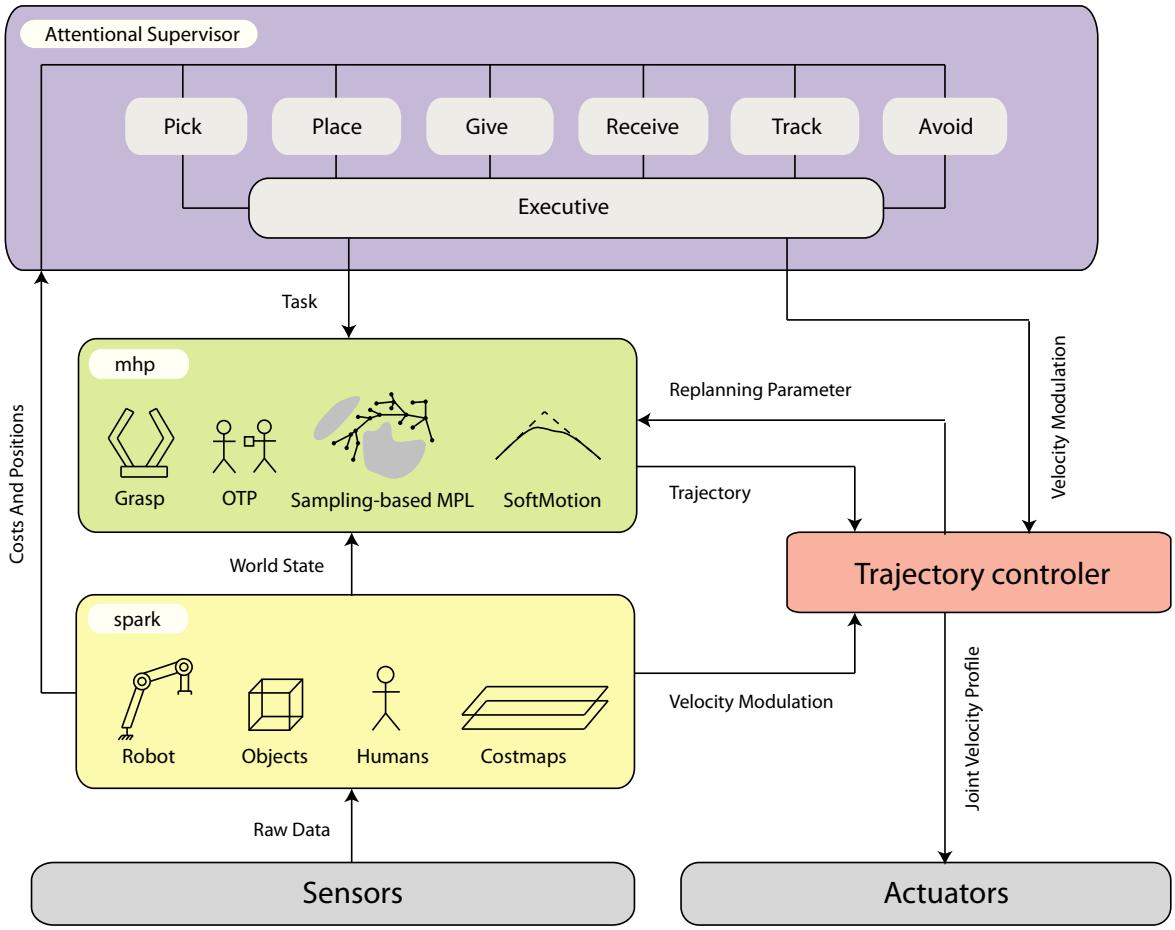


FIGURE 5.2 – Le système pour la manipulation interactive se décompose en 4 éléments : SPARK qui effectue un traitement initial des données capteurs, fournissant des cartes de coût, MHP qui planifie les tâches de manipulation, un contrôleur de trajectoire qui exécute la trajectoire produite par MHP et un système exécutif basé sur une approche comportementale.

5.1.1 Modèle de l'état du monde et mouvement

La Figure 5.2 détaille l'architecture logicielle mise en oeuvre pour la manipulation interactive. Pour le besoin des expériences nous avons intégré un système de perception des humains dans la scène basé sur le Kinect de Microsoft ainsi que les algorithmes de planification de mouvement et de calcul des coûts liés à l'interaction homme-robot. Le système de raisonnement spatial (SPARK) maintient un modèle géométrique du monde en traitant directement les données capteur. Il détermine la configuration géométrique du robot ainsi que la position des humains et des objets dans la scène. Les contraintes d'interactions sont également évaluées dans ce module par le biais de cartes de coût qui sont ensuite utilisées par le système attentionnel et le contrôleur de trajectoire ainsi que par le planificateur de mouvement.

A partir de l'état du monde, le planificateur de mouvement de manipulation (MHP) est

capable de proposer des trajectoires sûres et agréables pour des tâches de saisie et d'échanges d'objets. Le contrôleur de trajectoire est utilisé pour contrôler l'exécution des trajectoires de manipulation fournies par le planificateur MHP, il prend en compte les contraintes d'interaction pour moduler la vitesse. Il est également commandé par le système attentionnel pour assurer la sécurité et le confort de l'humain.

5.1.2 Supervision et tâches de manipulation interactives

A partir de l'évaluation des cartes de coût, de la posture et du comportement de l'homme dans SPARK, le système attentionnel peut moduler les fréquences d'échantillonnage de la perception et l'activation des actions (voir Figure 5.2). En fonction de seuils appropriés, le système d'exécution peut changer la tâche courante en sélectionnant une des tâches de manipulation telle que saisir, placer, donner ou recevoir ce qui implique un mouvement du bras et un actionnement de la pince. MHP décompose la planification d'une tâche de manipulation en deux étapes. Dans une première étape la configuration but est déterminée, puis dans une deuxième étape une trajectoire est planifiée à partir de la configuration courante. Quand la tâche change, le système exécutif est capable d'arrêter le mouvement et de lancer une nouvelle planification. Le système attentionnel module également la vitesse le long de la trajectoire exécutée.

5.2 Planification de mouvement pour la manipulation interactive

Dans un contexte interactif intégré le module de planification de mouvement doit être capable de générer des mouvements pour des tâches de manipulation de manière autonome. En plus de produire des chemins qui tiennent compte de certaines contraintes géométriques comme la distance à l'homme, il doit déterminer les configurations but et produire des trajectoires respectant certaines caractéristiques liées à leurs exécutions sur le robot. De plus il doit être capable de tenir compte de la dynamique introduite par la présence de l'homme ce qui nécessite un couplage entre la planification et l'exécution.

Le planificateur de mouvement présenté dans le chapitre 3 considère la géométrie mais ne tient pas compte des mouvements que peuvent effectuer les humains dans la scène. Dans la littérature relative à la planification de mouvement, différentes méthodes ont été introduites pour prendre en compte des changements dans l'espace de travail [Brock 00, Ferguson 06b, Zucker 07, Jaillet 08b]. Cependant, les obstacles mobiles pris en compte dans ces travaux ne présentent pas les mêmes contraintes que les humains. Pendant l'exécution de la trajectoire, l'humain peut venir proche du robot comme montré sur la Figure 5.1 et invalider les contraintes de sécurité et de lisibilité prises en compte pendant la planification du mouvement. Dans les situations de transfert d'objet prises en compte, l'humain peut également vouloir changer le point de transfert d'objet rendant la configuration but inappropriée pour la tâche.

Concernant l'exécution, le mouvement limité en jerk ainsi qu'en accélération et vitesse a été montré comme essentiel par Flash et Hogan [Flash 85]. Ils ont montré que le mouvement de l'homme est naturellement limité en jerk et accélération. De plus, les études d'échanges d'objets

de [Huber 08] suggèrent que les mouvements du robot avec un profil de jerk minimum de l'effecteur final conduisent à des échanges plus fluides. Dans [Broquère 10, Broquère 08], Broquère et al. ont introduit un système de génération de profils permettant de limiter le jerk, l'accélération et la vitesse pour assurer la sécurité de l'humain (vitesse) et le confort (jerk et accélération).

Dans ce chapitre, nous présentons le système de planification et de génération de mouvements pour la manipulation interactive avec replanification (section 5.2.1), ainsi qu'un système permettant de contrôler la vitesse d'exécution d'une trajectoire (section 5.2.2). Enfin, nous présentons une méthode permettant d'établir un lien entre planification géométrique et exécution qui garantit la conservation des contraintes en jerk, accélération et vitesse lors de l'exécution.

Contraintes géométriques pour l'interaction homme-robot

Les composants SPARK et MHP reposent sur une représentation de l'interaction sous forme de cartes de coût introduites dans [Sisbot 07a, Sisbot 07b] détaillée précédement. Ces cartes de coût permettent d'évaluer différentes propriétés comme le danger que peut induire la présence du robot dans certaines parties de l'espace de travail. Ainsi cette représentation permet d'évaluer les configurations qui peuvent rentrer en collision avec l'humain ou provoquer une gène soit parce quelles sont trop proches de l'humain soit parce qu'elles génèrent un effet de surprise en évaluant la distance et la visibilité de l'humain.

La carte de coût associée à l'accessibilité permet d'évaluer les parties de l'espace de travail qui sont accessibles à l'homme, notamment pour planifier des mouvements d'échanges d'objets. Ces propriétés sont prises en compte en tant que contraintes dans le cadre de la planification de mouvement mais sont également évaluées par le module raisonnement géométrique SPARK et accessibles à chaque instant au module de supervision. Dans ce cas, les cartes de coût peuvent être humano-centriques aussi bien que robot-centriques. Elles fournissent aux modules de plus haut niveau des informations telles que la distance à certains objets, l'accessibilité de ces objets ou la présence de l'humain dans les différents espaces d'interaction (intime, interaction, social).

5.2.1 Un planificateur de mouvement qui prend explicitement en compte les humains

Le planificateur de mouvement de manipulation interactive doit permettre de calculer des mouvements pour saisir, transporter et échanger des objets. Ici, nous considérons un système de supervision qui permet de sélectionner la tâche que doit accomplir le robot. Nous le détaillerons à la fin du chapitre.

En fonction de la tâche considérée, la configuration but est déterminée par une méthode qui calcule une configuration de saisie ou une position d'échange pour l'objet. Un chemin est alors planifié considérant les obstacles et les humains dans l'espace de travail entre la configuration courante et la configuration but. Ce chemin est ensuite traité par une étape de génération de trajectoire qui consiste à lui associer une loi de commande. La trajectoire obtenue respecte des contraintes dynamiques en fixant des limites en jerk, accélération et vitesse. Ces contraintes sont imposées simultanément par le système physique et l'interaction avec l'homme. D'autres approches spécifient la loi d'évolution temporelle en ligne, au niveau du contrôleur, l'intérêt

de traiter la génération de trajectoire au niveau du planificateur est de prendre en compte les collisions avec l'environnement afin de générer un mouvement fin pour traiter des cas contraints. Cette section détaille ces différentes étapes.

Planification de 'prise'

Nous utilisons le planificateur de prise introduit dans [Saut 12]. Même pour des tâches simples comme celles de saisir, déplacer et échanger un objet avec l'humain le choix de la prise est contraint par l'environnent ainsi que par la stabilité de la prise [Bounab 10]. La planification de prise consiste à trouver une configuration pour le préhenseur qui permette de saisir l'objet tout en satisfaisant un ensemble de contraintes.

Pour construire une liste de prises, le planificateur procède en trois étapes. Tout d'abord les situations possibles du préhenseur, la surface de l'objet, et l'espace de travail de chaque doigt sont discrétisés. Ces discrétisations sont importantes pour assurer une représentation la plus complète possible de l'ensemble des solutions. Les situations échantillonnées du préhenseur doivent notamment permettre de saisir l'objet par l'ensemble de ses éléments (i.e. dans le cas d'une tasse les solutions retenues doivent également permettre de saisir la anse et l'intérieur de la tasse). Après la discrétisation homogène de la surface de l'objet une hiérarchie de boîtes englobantes est construite pour calculer rapidement l'intersection avec le volume de travail des doigts. Ce volume est représenté par un ensemble de sphères. La deuxième étape permet de générer une liste de prises uniformément réparties autour de l'objet. Cette liste est ensuite filtrée pour sélectionner les prises stables et sans collisions. La longueur de cette liste correspond à un compromis afin de ne pas nuire aux performances de recherche d'une configuration complète de saisie tout en assurant de trouver une solution si il en existe une.

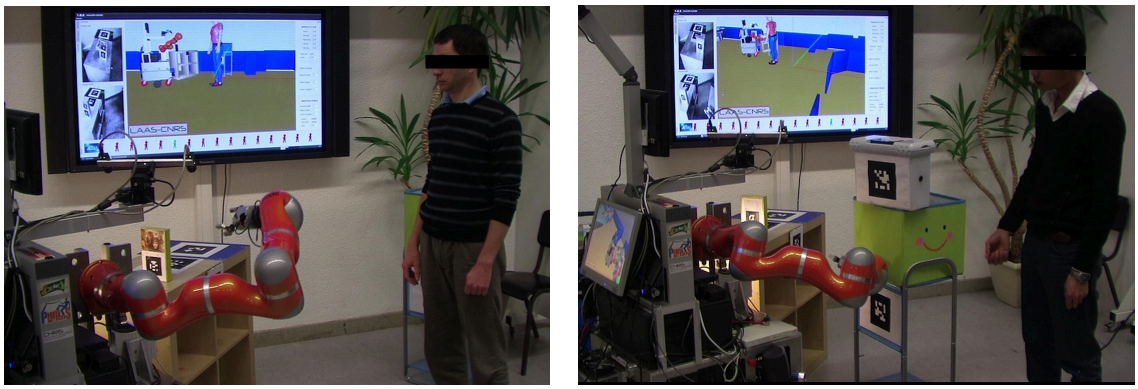
Le calcul de la configuration complète de saisie est effectué en parcourant la liste à la recherche d'une configuration sans collision. La cinématique inverse du bras manipulateur est utilisée. Un bras classique (6 DoFs) compte 8 classes de solutions. De plus pour un bras à sept degrés de liberté comme celui de Jido, le septième degré n'est pas déterminé par la méthode de cinématique inverse analytique. Il est échantillonné aléatoirement jusqu'à aboutir à une configuration sans collision ou atteindre un nombre limite d'essais pour une prise donnée. La Figure 5.3 présente deux solutions calculées pour un objet rectangulaire dans un espace de travail plus ou moins contraint.

Position de transfert d'objet (OTP)

La méthode retenue ici pour calculer un point de transfert parcourt la grille d'accessibilité de l'homme pour choisir une position appropriée de transfert d'objet. Cette grille intègre des coûts qui caractérisent la difficulté d'accès d'une cellule à partir de la posture courante. Elle est générée à partir d'une fonction évaluant le coût musculo-squelettique. Le coût d'une cellule est combiné aux coûts de distance et de visibilité pour obtenir un point satisfaisant au mieux ces trois contraintes. La configuration but est calculée de manière similaire au calcul des configurations de saisies en utilisant la cinématique inverse du bras du robot.

(a) Prise facile (b) Prise difficile

FIGURE 5.3 – Une prise facile (a) et une prise difficile (b) calculées en fonction des obstacles dans l'espace de travail. Le planificateur pour la manipulation est capable de sélectionner différentes prises en fonction du niveau d'encombrement dans l'espace de travail.



(a) Uncluttered Environment (b) Cluttered Environment

FIGURE 5.4 – Le planificateur de chemin est capable de traiter des environnements contraints par des obstacles.

Planification de chemin

Après avoir déterminé la configuration, un chemin est planifié à partir de la configuration courante. Nous employons le planificateur de mouvement détaillé dans le chapitre 3 qui prend en compte les contraintes d'interactions à l'aide des cartes de coût. Dans un premier temps le planificateur explore la carte de coût en utilisant la méthode T-RRT, puis dans un deuxième temps il effectue une optimisation locale pour améliorer le chemin. Cette méthode permet de traiter des problèmes de planification dans des environnements contraints comme illustré sur la Figure 5.4. Le chemin généré consiste en une succession de points étapes (configuration du robot) $\{q_{init}, q_1, q_2, \ldots, q_{target}\}$ connectés par des éléments de ligne droite.

Génération de trajectoire

A partir du chemin optimisé, une trajectoire est générée en utilisant la méthode [Broquère 10, Broquère 08]. Ce planificateur de trajectoire limite l'évolution en vitesse, accélération et jerk sur chaque axe. Les trajectoires sont représentées par des séries de courbes polynomiales du troisième ordre au plus. Les segments parcourus à vitesse (resp. accélération, jerk) constante sont d'ordre 1 (resp. 2, 3). Dans le cas multi-axes, sans perte de généralité, les instants de changement de courbe sont synchronisés. Ce modèle est approprié à la génération de trajectoire respectant des limites en vitesse, accélération et jerk. L'algorithme correspondant procède en deux étapes.

Dans un premier temps, une trajectoire TR_{ptp} point à point (ptp) est générée, composée d'une série de points étapes et d'une loi d'évolution qui s'arrête à chaque point étape. La trajectoire reliant deux points est calculée à partir de la projection des contraintes cinématiques sur l'axe joignant les deux points. Le résultat est une suite de sept segments composée de trois phases (accélération, vitesse constante, décélération). La phase d'accélération comprend un mouvement à jerk positif constant T_{jpa} suivi d'un mouvement à accélération constante T_{aca} (jerk nul) puis d'un mouvement à jerk négatif constant T_{jna}. La phase de décélération est similaire. Le segment du milieu parcouru à vitesse constante et nommé CVS_i pour le i^{eme} mouvement ptp, a une durée T_{vc}, qui peut être différente pour chaque mouvement. La durée de chaque segment est synchronisée pour tous les axes. La courbe représentée de couleur vert clair sur la Figure 5.5 représente le profil de vitesse de la trajectoire TR_{ptp} qui concatène les trajectoires reliant deux points.

La deuxième étape élimine les points étapes et lisse le chemin. Elle consiste à calculer des mouvements de transition pour chaque point étape q_i de la trajectoire TR_{ptp}. Ces mouvements de transition connectent la fin du segment à vitesse constante CVS_i (nommé M_i) avec le début du segment suivant CVS_{i+1}. La méthode proposée dans [Broquère 11] connecte ces deux points par une série de trois segments parcourus à jerk constant. Chaque mouvement de transition est vérifié avec le détecteur de collision. Dans le cas d'une collision, le système conserve la trajectoire initiale qui s'arrête au point étape q_i.

Les segments CVS_i de la trajectoire TR_{ptp} étant conservés par la trajectoire lissée TR, nous les utiliserons dans la section suivante pour établir un lien entre le chemin initial et la trajectoire réelle du robot. Ce lien permettra de synchroniser le processus de replanification. La trajectoire TR_{ptp} peut être modifiée de manière réactive en introduisant une fonction intermédiaire $\tau(t)$ qui module la loi de commande : $TR_{ptp}(\tau(t))$. Les contraintes HRI sont calculées à chaque instant par le module de raisonnement spatial pour déterminer la fonction $\tau(t)$.

5.2.2 Adaptation réactive de la vitesse et modifications du chemin

Jusqu'ici nous n'avons pas pris en compte de possibles changements pendant l'exécution de la trajectoire. Afin que le robot réponde aux mouvements humains, il est important de moduler la vitesse en influant sur la loi d'évolution $\tau(t)$ et de replanifier le mouvement du robot en fonction des contraintes d'interaction homme-robot. La Figure 5.2 qui décrit l'architecture globale de

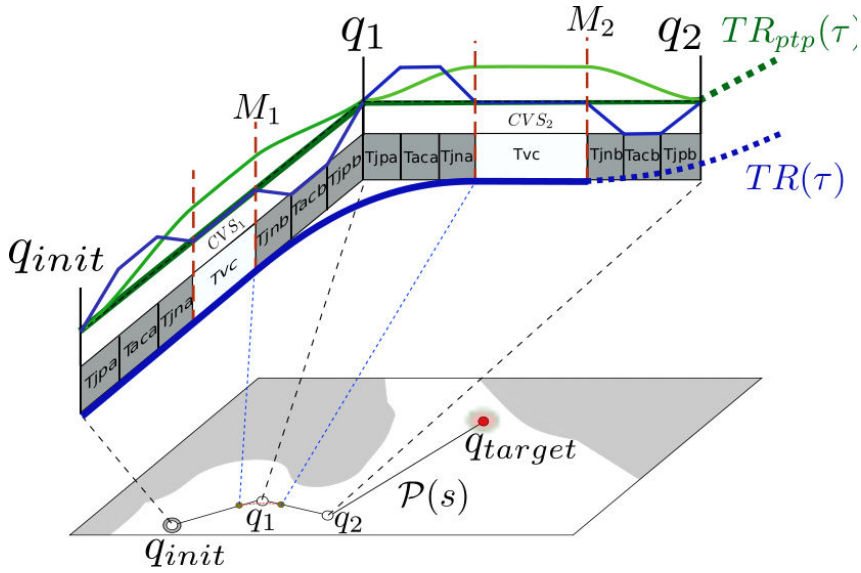


FIGURE 5.5 – La génération des trajectoires TR_{ptp} et TR. L'image du bas illustre un chemin $\mathcal{P}$ composé de lignes droites. Les courbes en haut sont les mouvements points à points (courbes d'accélération et vitesse entre chaque point). La courbe de couleur verte en gras représente la trajectoire TR_{ptp} et la courbe bleue en gras représente la trajectoire lisse TR.

ce système, présente les entrées du contrôleur de trajectoire qui viennent du module de raisonnement géométrique, du planificateur de mouvement et du système de supervision. Le contrôleur fournit également un paramètre de replanification qui indique l'instant courant d'exécution de la trajectoire au planificateur de mouvement.

Ainsi, le coût issu de l'interprétation géométrique des contraintes HRI modifie la loi temporelle $\tau(t)$ dans le contrôleur qui procure simultanément τ_c représentant la position courante du robot le long de TR afin de synchroniser le processus de replanification qui s'effectue dans un horizon de temps limité T_{Rep}.

Adaptation réactive de la vitesse en fonction des constraintes HRI

Les contraintes de distance et de champ de vision sont évaluées à chaque période de temps Δt au niveau du module raisonnement spatial. Notons que le contrôleur raisonne sur des zones de sécurité plus précisément par un calcul de distance exacte retournée par le détecteur de collision. Le coût de visibilité est évalué pour l'effecteur du robot. Le coût total $cost \in [0\ 1]$ est défini comme une somme pondérée des coûts de distance et de visibilité :

$$cost = \alpha * c_{dist} + \gamma * c_{visib}$$

Ce coût est nul quand aucun humain n'est détecté, le mouvement est alors exécuté à vitesse normale, définie lors de la génération de trajectoire. Si le robot pénètre dans la zone de sécurité, le coût augmente et le mouvement est ralenti, jusqu'à s'arrêter si la distance dépasse un certain seuil (5 cm). De manière similaire, quand l'humain détourne le regard, le robot ralentit jusqu'à

s'arrêter.

Pour avoir une vitesse normale quand le coût est nul et une vitesse nulle quand le coût est élevé, le contrôleur de trajectoire raisonne sur l'inverse du paramètre de coût $invCost(t) = 1 - cost$. Afin de maintenir les propriétés dynamiques de $invCost(t)$, on peut utiliser la méthode de lissage introduite dans [Broquère 10]. La fonction du temps $\alpha(t)$ représente la fonction $invCost(t)$ lissée. La fonction $\alpha(t)$ est mise à jour pour chaque pas d'échantillonnage (e.g. période) Δt du contrôleur de trajectoire et est utilisée pour adapter la loi de commande $\tau(t)$ le long de la trajectoire comme suit :

$$\begin{aligned} \tau(0) &= 0 \\ \tau(t) &= \tau(t - \Delta t) + \alpha(t)\Delta t \end{aligned} \tag{5.1}$$

Notons que lorsque le coût est nul, nous avons $\alpha(t) = 1$ et $\tau(t) = t$. La fonction $\alpha(t)$ est analogue à la vitesse d'évolution de $\tau(t)$. Cette méthode adapte la loi d'évolution pour toutes les articulations qui sont ralenties de manière synchronisée. Par exemple, dans le cas d'un mouvement à deux bras, les deux bras sont ralentis de manière synchrone.

La modulation de la vitesse limite le danger que représente une collision entre le robot et l'humain. Ce comportement, indispensable du point de vue de la sécurité n'est pas suffisant au regard de la lisibilité et du confort du mouvement exécuté. Dans la section suivante nous présentons une approche de replanificiation qui modifie le chemin initial afin de prendre en compte les mouvements de l'humain dans l'espace de travail.

Modification en ligne du chemin

La dynamique introduite par les humains peut invalider la trajectoire initialement calculée. Pour tenir compte de l'évolution de l'environnent, le système peut modifier le chemin initial et mettre à jour la configuration but ou replanifier le chemin $\mathcal{P}$ pour mieux prendre en compte les contraintes associées à la présence de l'humain. Le cas d'une configuration but invalide peut être trouvé, notamment dans les situations de transfert d'objets. La replanification de la configuration but implique une replanification de l'intégralité de la trajectoire. Nous traitons ici du cas de la replanification avec un horizon de temps T_{Rep} défini manuellement qui permet d'établir un lien entre la trajectoire exécutée et le chemin. Ce lien est discuté ultérieurement. Ce temps doit comprendre le temps imparti à la replanification et le temps de communication entre les modules de planification et de contrôle de la trajectoire.

Modification de la configuration but

Une nouvelle configuration but peut être calculée dans le cas d'un transfert d'objets. Dans ce cas, tous les segments se situant après les points étapes sont testés afin de raccorder la solution courante à la nouvelle configuration but (voir Figure 5.6). Afin de ne pas dégrader la solution initiale, les coûts associés aux segments (e.g. intégrale de coût) sont calculés et le segment valide minimisant les contraintes HRI est conservé. Le chemin est ensuite modifié en reconnectant la

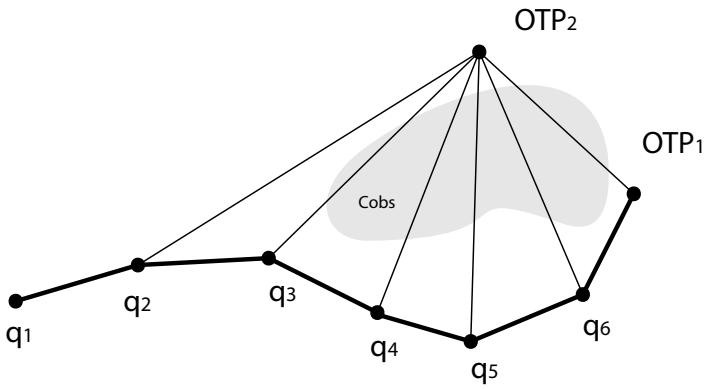
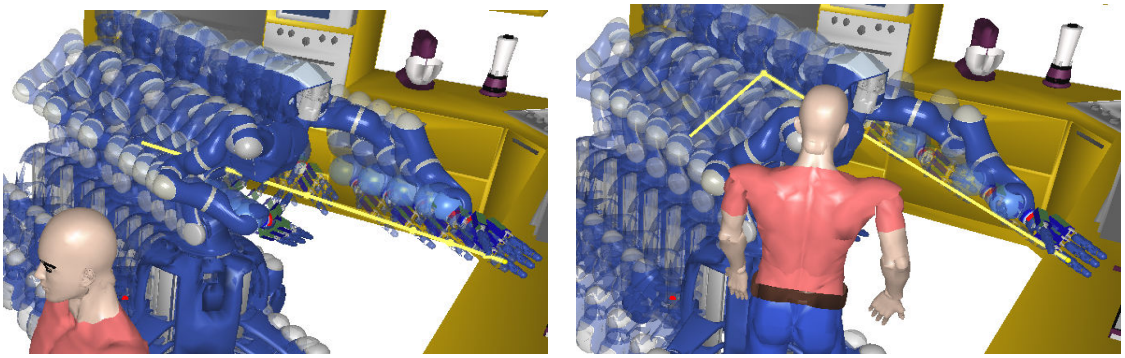


FIGURE 5.6 – Procédure d'ajustement de la configuration de transfert (OTP) en connectant la nouvelle configuration par parcour des points étapes.

ligne droite au point étape sélectionné et la fin du chemin est supprimé.

Déformation du chemin

Ensuite, si il n'y a aucune nouvelle configuration but ou si l'horizon de replanification n'est pas terminé, une méthode de déformation de chemin est utilisée. Cette phase d'optimisation permet d'améliorer le chemin en fonction des contraintes HRI afin de le rendre plus sûr et plus lisible. L'optimisation permet donc de prendre en compte l'état courant de l'humain qui peut avoir changé de position pendant l'exécution de la trajectoire (posture ou regard).



(a) Chemin initial (b) Nouveau chemin déformé

FIGURE 5.7 – Le modèle de Justin dans une tâche de saisie, le chemin de l'effecteur gauche est dessiné en jaune. Le chemin initial est déformé (la configuration finale reste la même) avec 5 secondes de la méthode de perturbation présentée au chapitre 3 pour mieux prendre en compte la distance et la visibilité.

La Figure 5.7(a) montre le robot Justin [Ott 06] qui comprend 13 DoFs actifs pour ce mouvement. La méthode de déformation de chemin est utilisée pour le mouvement de la base, du torse et du bras gauche. Un temps de replanification de 5 sec est utilisé, ce qui est raisonnable sachant que le robot doit parcourir 2 m de navigation à une vitesse de 0,4 m/s. La valeur des fonctions

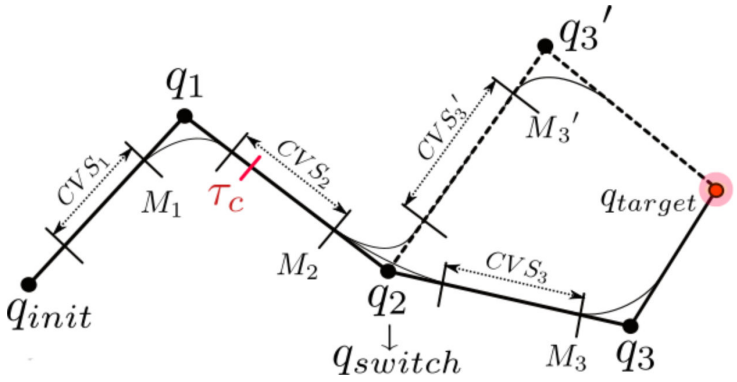


FIGURE 5.8 – Une itération de la méthode de déformation suivie par la technique de lissage de trajectoire est montrée. M_i sont les extrémités des segments CVS_i de la trajectoire TR, τ_c est la position courante du robot sur TR. Les lignes pointillées représentent le chemin modifié.

de coût associées aux contraintes HRI considérées par le planificateur de chemin diffère de celle prise en compte pendant la planification initiale. Ceci est dû au déplacement de l'humain dans la scène. Le nouveau chemin tient mieux compte de la proximité de l'humain en montant le bras et en s'écartant.

Selection du point étape pour modifier la trajectoire

Quand on exécute la trajectoire TR, l'horizon de temps T_{rep} et le paramètre courant τ_c donnent une configuration q_{switch} sur le chemin à partir de laquelle les modifications de la trajectoire sont autorisées. Compte tenu du lissage à l'étape de génération de la trajectoire, le chemin $\mathcal{P}$ diffère géométriquement de la trajectoire lissée TR (les points étapes sont supprimés par des mouvements de transition montrés sur la figure 5.8 [Broquère 08]). Nous proposons dans cette section une méthode pour déterminer l'ensemble des q_{switch} le long du chemin qui permettent d'effectuer une reconnection entre la nouvelle trajectoire et la trajectoire exécutée.

La Figure 5.8 présente un exemple de déformation de trajectoire. Dans cet exemple, le point étape q_3 est remplacé par $q_3\prime$ pendant la déformation du chemin. Modifier un point étape q_i implique des changements sur le chemin local correspondant. Comme introduit dans la section précédente, les seules parties du chemin $\mathcal{P}$ qui restent dans la trajectoire lissée TR sont les segments à vitesse constante (CSV). Chaque point sur les CSVs peut être utilisé pour reconnecter la trajectoire courante à une nouvelle trajectoire créée par modification du chemin $\mathcal{P}$. Le dernier point du i^{ime} segment CVS, M_i est utilisé pour la reconnection afin de laisser un temps plus long pour la replanification.

Afin de déterminer quel point étape peut être modifié, on peut définir un ensemble de paires contenant l'index du chemin local et le paramètre de temps à la fin du segment CVS comme suit :

$$M_i = \{i, \tau_i\} \tag{5.2}$$

Une liste de ce type de paires $M = \{M_1, M_2, ..., M_n\}$ est créée où n est le nombre de segments

composant la trajectoire. q_{switch} est sélectionné sur la trajectoire en choisissant la paire M_i solution de l'équation 5.3, où τ_c est le paramètre sur la trajectoire courante.

$$\min_{\tau_i} (\tau_c + T_{Rep} < \tau_i) \tag{5.3}$$

τ_i est donc le paramètre sur la trajectoire incluant un temps T_{Rep} entre τ_i et τ_c. Lorsque la nouvelle trajectoire est générée, elle remplace simplement la trajectoire exécutée sans discontinuité de jerk.

Ces différents composants permettent de planifier des trajectoires pour manipuler des objets dans des environnements encombrés et d'échanger des objets avec des humains. Les trajectoires exécutées sont limitées en jerk, accélération et vitesse et produisent donc un comportement souple et agréable pour l'humain. Enfin, l'adaptation de la vitesse et la replanification permettent de prendre en compte les mouvements de l'homme de manière réactive. La section 5.4 présente des résultats évaluant les techniques de replanification de chemin et d'adaptation de la vitesse en ligne.

La section suivante présente un système d'exécution qui élabore des comportements simples à partir des données produites par le système de raisonnement spatial SPARK. Ce système d'exécution est capable de sélectionner une tâche pour le planificateur de mouvement et de moduler la vitesse d'exécution en fonction du contexte d'interaction.

5.3 Module Attentionnel

Cette section présente un superviseur développé par notre partenaire UNINA et mis en oeuvre sur la plateforme Jido au LAAS.

Les tâches de manipulation interactive, en particulier l'échange d'objets, représentent une capacité de base qui est difficile à traiter [Edsinger 07, Sisbot 08a]. Les tâches élémentaires simples de transfert d'objet posent le problème de la coordination proche et continue entre les humains et les robots qui doivent surveiller et interpréter l'interaction en cherchant des opportunités tout en évitant les situations dangereuses.

Pendant la manipulation interactive, la coordination sensori-motrice est nécessaire. C'est la raison pour laquelle les mécanismes attentionnels [Norman 86, Kahneman 73, Posner 75] devraient jouer un rôle crucial. Ces mécanismes devraient : diriger les capteurs vers la source la plus significative d'information afin de rapidement réagir à des mouvements importants de l'humain, filtrer les données sensorielles disponibles et fournir des mécanismes de coordination pour orchestrer et fixer les priorités dans le cas d'activités concurrentes et coopératives. Les mécanismes attentionnels en HRI ont été principalement étudiés en se concentrant sur l'attention visuelle conjointe [Nagai 03, Lang 03, Breazeal 02, Breazeal 05, Trafton 05, Kaplan 06]. Dans ces travaux, les auteurs ont introduit et analysé des mécanismes d'attention conjointe (regard, pointage) jouant un rôle de communication implicite utilisé pour améliorer la qualité de l'interaction sociale homme-robot. Par ailleurs, des travaux antérieurs ont proposé le développement d'un superviseur attentionnel [Norman 86, Cooper 00] adapté pour traiter l'interaction homme-robot

de manière sûre et efficace pendant la manipulation.

5.3.1 Architecture attentionnelle pour la manipulation interactive

Le système attentionnel considéré ici est un système réactif, "behavior-based", doté de mécanismes attentionnels "bottom-up" capables de surveiller plusieurs processus de manière concurrente [Kahneman 73]. En particulier, le système utilise un modèle d'allocation de l'attention basé sur la fréquence [Senders 64, Burattini 08, Burattini 10]. Suivant cette approche, le mécanisme d'allocation attentionnnel permet de réguler et de distribuer la résolution à laquelle différents processus concurrents liés à la manipulation interactive sont surveillés et contrôlés par modulation de la fréquence correspondante.

Le système considère des processus tels que saisir, déplacer, donner et recevoir ainsi que des comportements liés à la perception de l'environnement. Le mécanisme d'allocation attentionnel prend en compte les dispositions et activités de l'humain pour équilibrer la sécurité et l'exécution de la tâche. L'état de l'interaction entre l'homme et le robot est évalué par les cartes de coût dans le module de raisonnement spatial SPARK. c_{dist}, c_{vsisib} et c_{reach} peuvent être humano-centrique ou robot-centrique et correspondent aux coûts liés à la distance, au champ de vision et à l'accessibilité. Ces représentations procurent donc une évaluation uniforme de l'état de l'interaction homme-robot partagée par le système attentionnel, le planificateur et le contrôleur de trajectoire. En plus des cartes de coût, le système de raisonnement spatial procure un ensemble de données au système attentionnel comme la position des objets et leurs vitesses (pos_o et vel_o où o est l'identifiant de l'objet), l'état de la pince (ouvert ou fermé) ou la distance entre la pince et un objet donné (d_{go}) qui sont combinées pour élaborer les fréquences d'activation des différents processus.

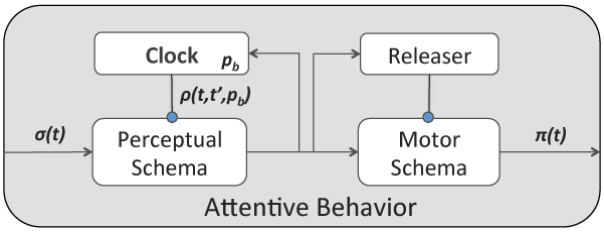


FIGURE 5.9 – Schéma de representation d'un comportement attentionnel.

Le modèle attentionnel basé sur la fréquence est représenté sur la Figure 5.9 par un *schéma théorique* [Arbib 98]. Il se caractérise par un *schéma perceptuel* (SP) qui élabore les données capteur $\sigma(t)$, et un *schema moteur* (SM) produisant le modèle des actions motrices ainsi qu'un *releaser* [Tinbergen 51] fonctionnant comme une déclencheur pour l'activation du SM. Le mécanisme de contrôle d'attention est basé sur une *horloge* qui régule la fréquence d'échantillonnage de la perception et l'activation des comportements. L'horloge régule la résolution à laquelle le comportement est surveillé et contrôlé. De plus, elle procure un critère de priorité simple. En effet un comportement associé à une fréquence élevée sera considéré comme prioritaire par rapport à

un comportement associé à une fréquence faible.

Mise en oeuvre des comportements

L'architecture attentionnelle de la Figure 5.10 intègre des comportements pour saisir, déplacer, donner et recevoir, ainsi que pour chercher et traquer les humains et les objets. La régulation des comportements comme l'attitude d'évitement y est aussi représentée. Nous donnons ici une vue globale du fonctionnement du système représenté sur la Figure 5.10 sans donner les détails des fonctions intermédiaires de calcul des fréquences.

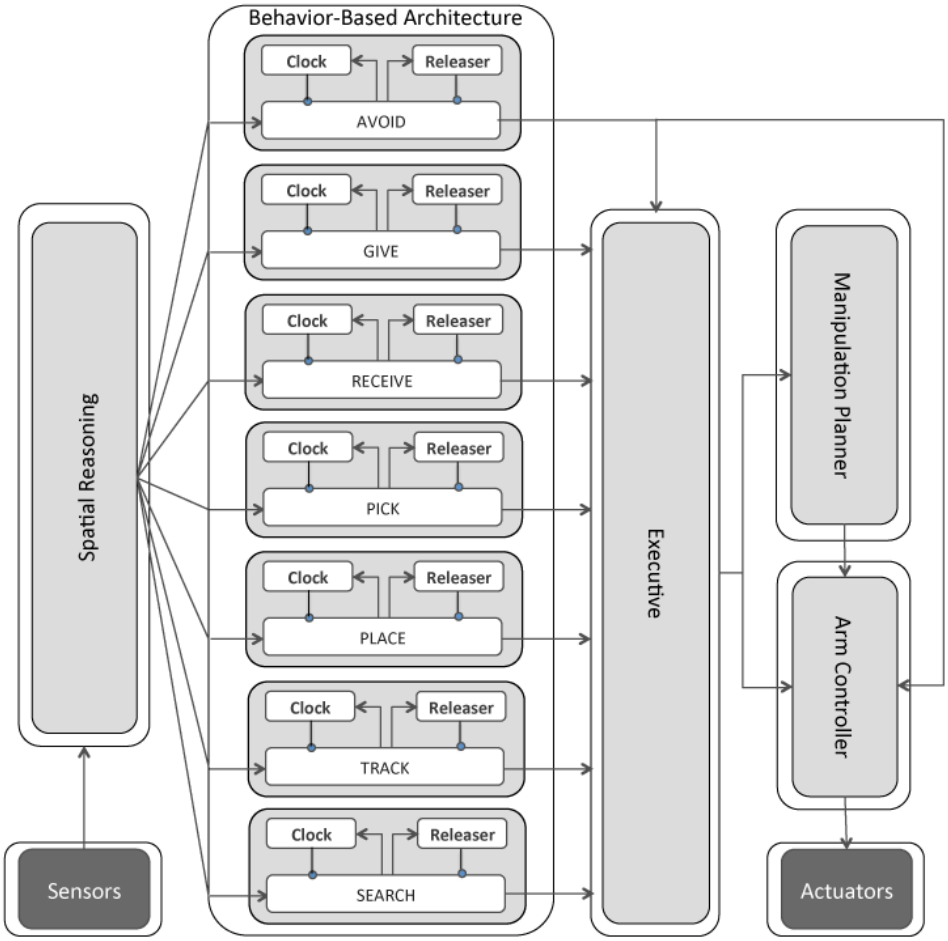


FIGURE 5.10 – Le système attentionnel comportemental dans l'architecture globale. Le système de raisonnement spatial procure au système attentionnel des données pré-traitées et influence le changement de tâches (exécutif) et le contrôle du bras (contrôleur de trajectoire).

`SEARCH/RECHERCHER` définit l'action de scruter l'environnent à la recherche d'objets et d'humains. Le signal surveillé est la distance du bassin de l'humain au robot ($\sigma_{tr}(t)$). Ce comportement est activé quand le système robotique est peu sollicité et les objets ou les humains sont dans le champ de vision du robot. Lorsque l'humain est détecté dans l'espace de travail lointain du robot (c-à-d. quand $3m < \sigma_{tr}(t) \leq 5m$), la tâche `TRACK/SUIVI` est activée et permet au

robot de surveiller les mouvements des humains avant qu'ils n'entrent dans l'espace d'interaction ($1m < \sigma_{tr}(t) \leq 3m$).

`AVOID/EVITER` supervise la sécurité de l'humain pendant l'interaction. Ce comportement est activé lorsque l'humain est détecté dans l'espace d'interaction du robot ($1m < \sigma_{av1}(t) < 3m$). Il module la vitesse des mouvements du bras par rapport au comportement de l'humain et interrompt les mouvements du bras quand la situation est évaluée comme dangereuse. La distance minimale homme-robot ($\sigma_{av1}(t)$) est surveillée dans l'intervalle $0.1m < \sigma_{av1}(t) \leq 3m$. La fréquence de l'horloge dépend de la proximité de l'opérateur et de la visibilité du robot. Si l'opérateur est proche et/ou la distance homme-robot pour les mains et la tête augmente entre des lectures des capteurs successives, l'horloge est accélérée et elle est décélérée lorsque l'opérateur se déplace en s'écartant du robot. Si le robot n'est pas assez visible ou si il est dans l'espace de proximité `AVOID` arrête le robot en imposant une vitesse nulle.

`PICK/SAISIR` est activé quand le robot ne tient pas d'objet mais qu'il existe un objet atteignable par le robot. Le comportement est donc activé quand la distance d'un objet o, $\sigma_{pk1} = d_{go}$ à effecteur est $\sigma_{pk1} \leq 3m$ et quand le coût d'accessibilité σ_{pk2} est sous un seuil approprié ($\sigma_{pk2} < K_{reachability}$). Ce comportement commence avec un processus de planification qui génère une trajectoire vers l'objet donné. Ensuite une procédure de saisie est activée et lorsque le robot tient l'objet il le déplace vers une position sûre proche du corps du robot. `PLACE/POSER` est activé quand le robot tient un objet en l'absence d'humain dans l'espace interactif. Il procède de la même manière que `PICK` en planifiant une tâche de manipulation. Pour ce comportement l'effecteur du robot place l'objet et repart vers une position de repos. Dans ce cas, la période d'horloge est régulée par une fonction qui est analogue à celle de `PICK`.

`GIVE/DONNER` et `RECEIVE/RECEVOIR` régulent les activités de donner et recevoir des objets. Ils prennent en compte la position et les mouvements des humains dans l'espace de travail ainsi que les coûts d'accessibilité et de visibilité. `GIVE` est activé quand un humain se trouve dans l'espace d'interaction ($\sigma_{gv1} = c_{dist}(r,p)$ et $\sigma_{gv1} \leq 3m$), l'effecteur est visible, ($\sigma_{gv2} = c_{visib}(h,r)$ et $\sigma_{gv2} < K_{visibility}$), la main de l'humain est accessible ($\sigma_{gv3} = c_{reach}(h, hand)$ et $\sigma_{gv3} < K_{reachability}$) et le robot tient un objet. La période d'horloge est ici associée à la distance et la vitesse de l'humain. D'autre part, si plus d'une cible humaine est disponible, `GIVE` sélectionne celle qui a un coût minimal dans la carte de coût d'accessibilité. Après activation, ce comportement déplace l'effecteur vers le point de transfert en modulant la vitesse par rapport à la distance de la main de l'humain. Finalement, le comportement `RECEIVE` est activé quand l'humain est dans l'espace d'interaction ($\sigma_{gv1} = c_{dist}(r,p)$ et $\sigma_{gv1} \leq 3m$), porte un objet ($\sigma_{gv4} = d_{go}$ et $\sigma_{gv4} < dmax_{go}$), l'effecteur du robot est visible ($\sigma_{gv2} = c_{visib}(h,r)$ et $\sigma_{gv2} < K_{visibility}$) et les mains de l'humain sont accessibles ($\sigma_{gv3} = c_{reach}(h, hand)$ et $\sigma_{gv3} < K_{reachability}$). La fréquence d'échantillonnage est régulée par une fonction analogue à la période de `GIVE`.

L'ensemble de ces actions permettent de générer un large éventail de comportements pour échanger des objets avec des humains mais nécessite d'être interprété par une couche d'exécution.

5.3.2 Système d'exécution

Le système exécutif reçoit les données du système attentionnel et gère l'exécution des tâches orchestrant ainsi le planificateur de mouvement et le contrôleur de trajectoire. Il surveille de manière continue l'activation et la désactivation des différents comportements ainsi que les activités associées comme la fréquence des horloges. Il décide quand passer d'une tâche à une autre et quand arrêter ou interrompre l'exécution des tâches et comment moduler la vitesse d'exécution.

Initialement le système exécutif est au repos. Lorsqu'un comportement s'active dans le module attentionnel il change d'un état de repos à l'une des quatre tâches suivantes : saisir, poser, donner ou recevoir. Les comportements sont activés de manière concurrente, et peuvent être en conflit. Par exemple l'activation de `RECEIVE` peut être en conflit avec `PICK`, ou `GIVE` en conflit avec `PLACE`. Dans ce cas, le système exécutif reste engagé dans la tâche courante à moins que la fréquence associée au comportement qui est en conflit avec la tâche courante dépasse la fréquence de celui qui est exécuté par une valeur seuil convenable.

Afin d'activer une tâche le système exécutif doit non seulement sélectionner le comportement associé, mais également l'objet le plus approprié pour la manipulation et l'humain cible. Une tâche est donc définie par le triplet (*comportement*, *human*, *object*). Le système exécutif doit surveiller que le triplet associé reste actif pendant l'exécution, interrompre le mouvement courant et planifier un nouveau mouvement si l'une des trois composantes change.

Parallèlement, le système exécutif surveille le comportement `AVOID` pour éviter les collisions avec les objets et avec les humains. La modulation de la vitesse du bras est obtenue comme le minimum entre celle associée au comportement courant et celle suggérée par `AVOID` : $\alpha(t) = min(\alpha_{av}(t), \alpha_{task}(t))$. De plus, `AVOID` peut directement contourner le système exécutif (voir Figure 5.10) pour arrêter le mouvement dans les cas dangereux d'interactions.

Résumé

Nous avons présenté une architecture complète intégrant nos travaux sur la prise en compte de la dynamique introduite par l'homme dans la planification et l'exécution de chemin en interaction avec l'homme. Cette architecture logicielle intègre également les travaux sur la planification de chemin présentés dans le chapitre 3. Le passage à un système intégré nécessite de déterminer les configurations but et de coupler le planificateur au système d'exécution afin de replanifier quand nécessaire. Ce système est orchestré par une approche comportementale attentionnelle de nos partenaires d'UNINA à Naples en Italie.

5.4 Résultats

Pour illustrer l'approche nous présentons les résultats issus de simulations et d'expériences sur le robot Jido du LAAS :

- L'asservissement de la vitesse sur les contraintes HRI générées par SPARK.
- Une simulation de replanification pour mieux prendre en compte les contraintes HRI.

– L'évaluation du planificateur et du système attentionnel grâce à des expériences menées lors de deux séjours d'une semaine de travail au LAAS avec trois chercheurs de l'université. d'UNINA.

5.4.1 Plateforme expériementale

Jido est construit avec une plateforme Neobotix MP-L655, un bras Kuka LWR-IV et est équipé avec une paire de caméras stéréo. Un Kinect est utilisé pour suivre le corps de l'humain. Pour le regard, nous utilisons un système de capture de mouvement. L'architecture logicielle du système robotique est basée sur des modules Genom [Fleury 97].

Le module *SPARK* maintient le modèle 3D de l'environnement, le module *MHP* planifie les trajectoires qui sont exécutées par *LWR*, le module de contrôle de l'exécution de la trajectoire sur le bras Kuka. Le module *NIUT* est chargé de suivre la cinématique des humains et utilise le Kinect. Le module *MOCAP* (système de capture de mouvement) suit l'orientation de la tête des humains et utilise des marqueurs. Le module *VIMAN* traque les objets dans la scène grâce à un système d'étiquettes basé sur ARToolKit [Prince 02].

5.4.2 Modulation de la vitesse

Les Figures 5.11, 5.12 et 5.13 montrent trois exécutions de la même trajectoire à proximité de l'homme avec les profils de vitesse correspondants. Le comportement de l'humain affecte les profils de vitesse qui sont tracés pour chaque articulation du bras (méthode issue de [Broquère 11]). Sur la Figure 5.11 la trajectoire est exécutée sans mouvement de l'homme dans l'espace de travail. Les profils de vitesse sur chaque axe correspondent à ceux produits par la génération de trajectoire. Par contre sur la Figure 5.12, l'humain détourne le regard ce qui provoque un ralentissement puis un arrêt complet du robot. Enfin, sur la Figure 5.13, l'humain approche sa main jusqu'à atteindre le seuil de 5 cm à plusieurs reprises. Ceci provoque des arrêts répétés de l'exécution de manière synchrone pour toutes les articulations garantissant un comportement sûr.

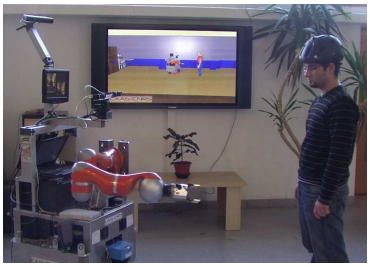
5.4.3 Replanification

La Figure 5.14 présente un chemin déformé par la méthode de perturbation présentée au chapitre 3 avec un horizon de 3 secondes et la fonction de coût paramétrique associé à cet instant. La Figure 5.14(a) montre un chemin initial calculé en tenant compte des contraintes HRI courantes. Sur la Figure 5.14(c) les contraintes ont changé, et sont prises en compte par la méthode de perturbation aléatoire de chemins (Figure 5.14(e)). Le nouveau chemin est plus court et minimise la distance et le coût sans dégrader le coût de visibilité.

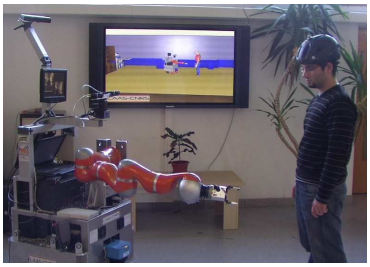
5.4.4 Planificateur de manipulation

Dans un premier test expérimental, nous avons voulu évaluer la performance de la planification en interaction avec l'homme et celle du système de contrôle dans des environnements

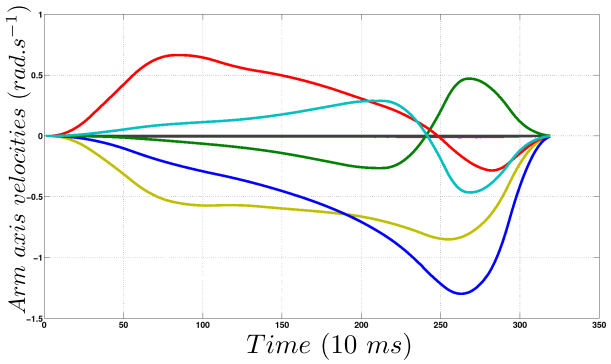
(a) Début



(b) Milieu



(c) Fin



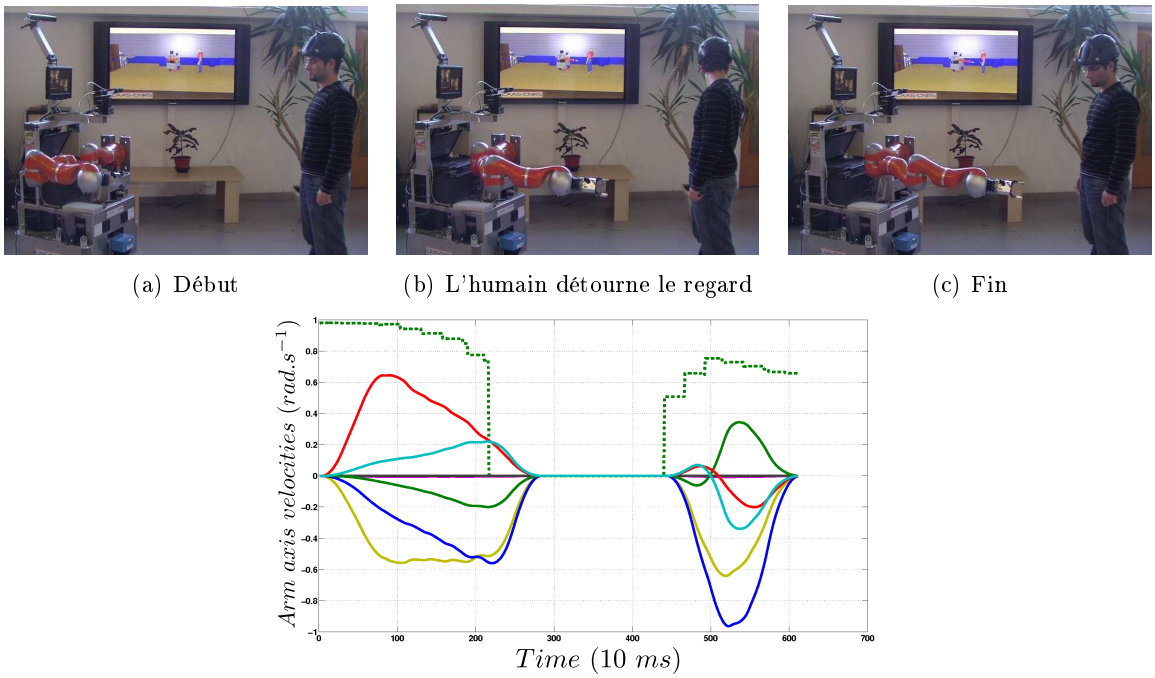
(d) Profil de vitesse sur les 7 axes du bras kuka

FIGURE 5.11 – Le robot exécute une trajectoire sans influence d'humain

contraints et non contraints (Figure 5.4 (a) et (b)). Les modèles CAO des objets dans l'environnement sont connus, les positions et orientations des objets et des obstacles dans l'environnement sont mis à jour en utilisant des caméras stéréo et des étiquettes chaque 25 ms. La position et la posture de l'homme est mise à jour en utilisant le Kinect.

La tâche consiste à saisir un objet sur une table et le donner à l'homme. Afin d'exécuter l'expérience en boucle, le processus de saisie est démarré dès que la paire de caméras stéréo détecte un objet sur la table. Lorsque le Kinect détecte un humain dans l'espace d'interaction, le planificateur calcule une trajectoire pour donner l'objet. Dès qu'une trajectoire valide est trouvée, le robot l'exécute et donne l'objet à l'humain. Dans cette expérience, nous collectons les temps de planification pour des tâches de saisie et de transfert d'objet ainsi que la durée de chaque trajectoire. Le tableau suivant (Table 5.1) présente les résultats pour un environnement encombré ou dégagé. L'environnent encombré est visible sur la Figure 5.15, l'environnement dégagé est identique mais ne contient pas les objets à la droite de l'humain. Ces données regroupent les résultats pour 53 essais. La version du planificateur évalué ne tient pas compte des changements dans l'environnent.

Pour les environnements dégagés, les temps de planification et d'exécution sont assez rapides (1.29 sec pour une saisie et 2.75 sec pour donner l'objet) et compatibles avec l'interaction homme-robot. Quand l'environnement devient plus encombré, les temps de planification augmentent (5.45 sec pour une saisie et 12.18 sec pour donner l'objet) et l'interaction manque de naturel et de réactivé.



(a) Début (b) L'humain détourne le regard (c) Fin

(d) Profil de vitesse

FIGURE 5.12 – Pendant que le robot exécute la trajectoire, l'humain proche du robot détourne son regard. Le robot ralenti puis s'arrête en attendant que l'humain regarde à nouveau.

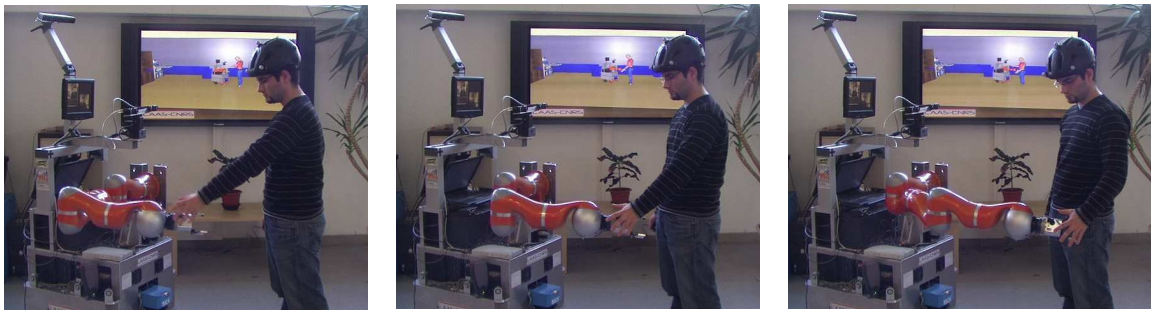
Durée	Saisie		Donner l'objet		Total
	planification	execution	planification	execution	
Moyenne	1.29s	6.61s	2.75s	10.20s	20.85s
Min	0.72s	5.00s	0.99s	5.58s	12.29s
Max	5.45s	24.52s	12.18s	22.34s	64.49s
STD	0.81s	3.51s	2.01s	3.75s	5.5s

TABLE 5.1 – Performances de planification et d'execution

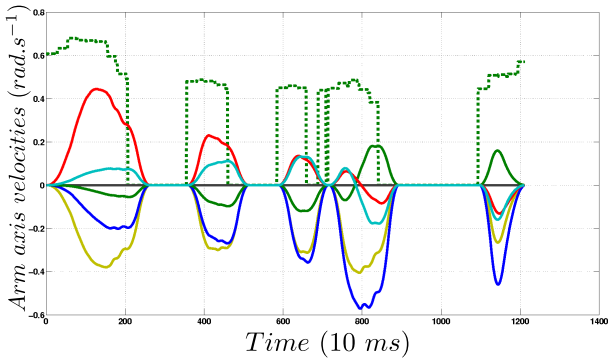
Système Attentionnel

Dans une deuxième expérience, nous avons testé le système attentionnel en considérant ses performances dans l'allocation, la sélection de la tâche et la modulation de la vitesse. A cet effet, nous définissons un deuxième scénario, dans lequel le robot doit orchestrer les tâches suivantes : saisir un objet sur la table, donner l'objet à l'humain, recevoir un objet de l'humain ou placer l'objet à une position prédéfinie. Dans ce scénario, en l'absence d'humain, le robot surveille la table et s'il détecte un objet, il effectue un mouvement de saisie avant de placer l'objet dans une boite posée sur le sol. Si l'humain vient pour donner un objet, le robot reçoit l'objet. Si un humain est prêt à recevoir un objet, le robot donne l'objet qu'il tient.

Sur la Figure 5.15 après avoir effectué la saisie d'une boîte à cassette sur la table, le robot trouve un chemin évitant les obstacles et donne l'objet à l'humain. Notons qu'une fois l'objet saisi, le chemin est planifié pour l'ensemble robot et boîte, évitant ainsi les collisions entre la



(a) Début (b) Milieu (c) Fin



(d) Profil de vitesse

FIGURE 5.13 – L'humain approche son bras plusieurs fois ce qui arrête systématiquement le robot le long du chemin.

cassette et l'environnement.

Dans ce contexte, nous évaluons les performances du système attentionnel en terme d'activation de comportement et de modulation de la vitesse. Le système attentionnel doit réduire les activations en les concentrant sur les situations pertinentes. De manière similaire, la vitesse doit être réduite seulement si nécessaire (c-à-d. dans les cas dangereux). Pour évaluer l'efficacité du système attentionnel, nous considérons le pourcentage d'activation des comportements (par rapport au temps total disponible) et la valeur moyenne de la fonction de modulation de vitesse $\alpha(t)$ pour chaque phase d'interaction. En particulier, pour chaque tâche nous analysons les activations de deux comportements : le comportement dominant pendant la tâche (celui qui caractérise la tâche, c-à-d. le comportement saisie pendant la tâche saisie) et le comportement 'avoid'. L'idée est que, le système attentionnel est efficace s'il peut réduire les activations sans affecter le taux de succès et la sécurité. De manière similaire, la valeur moyenne de la fonction de modulation de la vitesse $\alpha(t)$ doit être maximisée en préservant les taux de succès, de sécurité et de qualité de l'interaction.

Cinq sujets ont participé aux expériences. Chacun a réalisé 4 essais, donc nous avons effectué un total de vingt essais. Les sujets n'étaient pas informés du comportement du système. Nous avons évalué le système en considérant à la fois des mesures quantitatives et subjectives par l'utilisation d'un questionnaire rempli par les sujets après chaque test. Le tableau 5.2 présente les résultats obtenus à partir de l'évaluation des enregistrements vidéo associés au essais. Nous

	Saisir	Donner	Placer	Recevoir
Activations Comportement	0.28 ± 0.15	0.18 ± 0.04	0.27 ± 0.09	0.11 ± 0.05
Activations Evitement	0.26 ± 0.12	0.31 ± 0.15	0.61 ± 0.25	0.72 ± 0.25
$\alpha(t)$	0.49 ± 0.2	0.62 ± 0.24	0.45 ± 0.17	0.59 ± 0.2

TABLE 5.2 – Allocation attentionnelle pendant les différentes phases d'interaction (Saisir, Donner, Placer, Recevoir)

avons segmenté chaque action : saisir, placer, donner, recevoir. Les données collectées ont des valeurs moyennes et des écarts types (STDs) relatifs aux 20 essais.

A partir des résultats collectés, nous pouvons observer que pour chaque phase, le pourcentage d'activation des comportements dominants et des comportements d'évitement reste bas par rapport au temps total. Le système attentionnel est donc efficace pour réduire et concentrer les activations. Concernant la modulation de la vitesse, contrairement à ce qui est attendu, les valeurs pour $\alpha(t)$ semblent légèrement plus élevées pendant les phases de 'saisie'/'recevoir', probablement parce que dans notre réglage l'environnent est généralement plus encombré pour ce type d'action. De plus, comme observé précédemment par [Edsinger 07], si les mouvements du robot sont lisibles, le transfert d'objet est généralement facilité par le comportement collaboratif de l'humain, ainsi la valeur moyenne de la modulation de la vitesse reste haute.

Pour évaluer la durée de l'interaction et la fiabilité du système nous évaluons les enregistrements vidéos des 20 tests. Par ailleurs, la qualité de l'interaction a été évaluée en demandant aux sujets de remplir un questionnaire après chacun des 20 tests. Les événements liés au début du mouvement du robot et à la saisie de l'humain ou du robot sont utilisés pour déterminer la durée de l'interaction. Le questionnaire avait des entrées pour évaluer les caractéristiques suivantes : la sécurité (est-ce que l'interaction est sûre ?) ; le naturel (est-ce que l'interaction est naturelle ?) ; la lisibilité du comportement du robot (est-ce que l'humain comprend le comportement du robot ?). Chaque entrée pouvait être évaluée avec un score de 1 (très mauvais) à 10 (très bon).

Le tableau 5.3 et le tableau 5.4 présentent les résultats obtenus pour chaque phase d'interaction (saisir, placer, donner, recevoir) avec des analyses vidéos et le questionnaire. Dans le tableau 5.3, *temps* est le temps nécessaire pour réaliser la tâche dans son ensemble à partir de la sélection d'un comportement jusqu'à son succès ou son échec, alors que *échecs* est le taux d'échec par rapport au nombre de tentatives. Dans le tableau 5.4, les caractéristiques *sécurité*, *naturel*, *lisibilité* sont les moyennes et les écarts types des scores donnés par les sujets de l'expérience.

	Saisir	Donner	Placer	Recevoir
Temps	12.3 ± 6.3	14 ± 1.41	12.25 ± 3.41	15.8 ± 6.14
Echecs	0.1 ± 0.09	0.1 ± 0.1	0.09 ± 0.1	0.2 ± 0.12

TABLE 5.3 – Durée de l'interaction et fiabilité analysées à partir de l'évaluation des vidéos

Les valeurs dans le tableau 5.3 montrent que les temps pour atteindre le but sont assez stables pour les comportements qui nécessitent la coopération de l'humain. En comparant les tableaux

	Saisir	Donner	Placer	Recevoir
Sécurité	10 ± 0	9.8 ± 0.3	8.2 ± 1.3	7.2 ± 1
Naturel	9 ± 1.2	8.6 ± 1.8	8 ± 1.5	7.1 ± 0.88
Lisibilité robot	9.4 ± 1.3	9.6 ± 0.8	9.3 ± 0.7	6 ± 2.68

TABLE 5.4 – Analyse quantitative du questionnaire d'évaluation

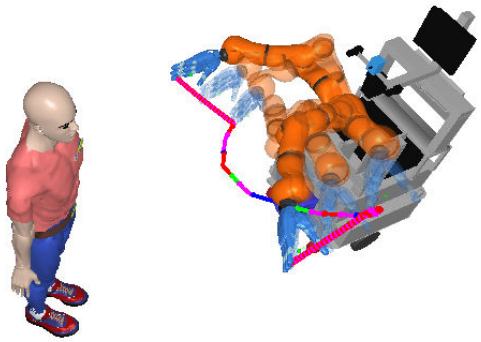
5.3 et 5.4, nous observons que la tâche est fiable pour chaque phase. Comme attendu, la valeur de la sécurité est haute pendant les 'saisir' et 'placer'. Généralement l'humain est loin du robot quand les opérations 'saisir' et 'placer' sont activées. Cependant la valeur de la sécurité reste relativement haute pendant les phases de 'donner' et de 'recevoir'. Ceci est certainement dû au fait que le mouvement du robot est lent et que les participants sont situés à une distance élevée du robot. Pour la tâche 'recevoir', l'interaction est évaluée comme moins sûre, moins naturelle et moins lisible. Ceci est dû aux mouvements générés pour ce type de tâche qui sont grands et longs comme nous pouvons le voir dans la durée d'interaction du tableau 5.3.

5.5 Conclusions

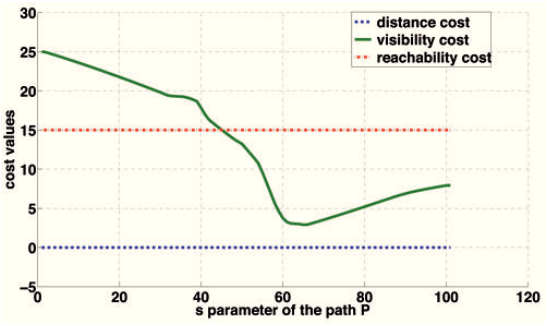
Nous avons présenté une architecture adaptée à la manipulation collaborative, sûre et efficace dans le cadre de l'interaction homme-robot. Le système proposé intègre un contrôleur de trajectoire et un système de supervision attentionnel avec le planificateur de mouvement en interaction avec l'homme. Les représentations par cartes de coût qui modélisent les contraintes d'interaction sont partagées par les différents composants du système afin d'évaluer le danger, de déterminer les opportunités d'interaction et d'augmenter le confort de l'humain.

Le planificateur de mouvement prend en entrée la prise ou le point de transfert et produit des chemins qui respectent les contraintes géométriques. Des trajectoires limitées en jerk, accélération et vitesse sont ensuite générées à partir de ce chemin. Son intégration a été facilitée par la présence du module de raisonnement spatial qui fournit un état du monde issu des données capteur. Nous avons intégré le Kinect au module de raisonnement spatial pour ces expériences, ce qui a enrichi l'architecture déjà présente au laboratoire. Ce travail a été effectuée en collaboration avec différentes personnes au LAAS dont Xavier Broquère, Jean Philipe Saut et Moktar Gharbi.

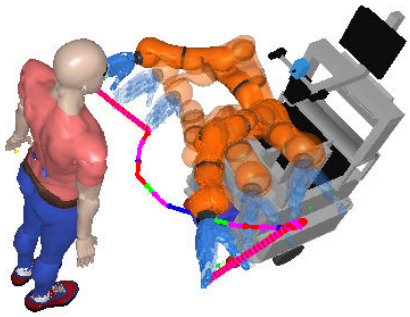
Le système présenté intègre également nos contributions pour la prise en compte de la dynamique introduite par l'homme. Elles concernent notamment l'adaptation de vitesse, la déformation de chemins et un lien entre le chemin et la trajectoire limité en vitesse, accélération et jerk afin de mieux tenir compte des contraintes HRI pendant l'exécution du mouvement. Nous avons détaillé l'étude utilisateur évaluant la performance du système global. Les résultats collectés montrent que le système attentionnel d'UNINA qui régule les différents mécanismes, est réactif et bien adapté au système de contrôle pour assurer la sécurité la fiabilité et l'efficacité mais également le "naturel" et la lisibilité de l'interaction.



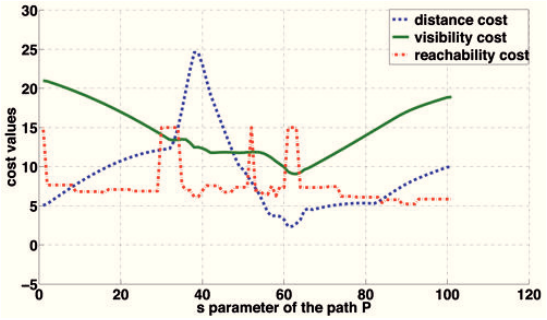
(a) Chemin inital, Contraintes initiales



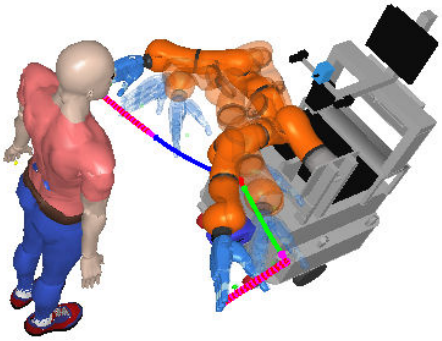
(b) Profil de coût (a)



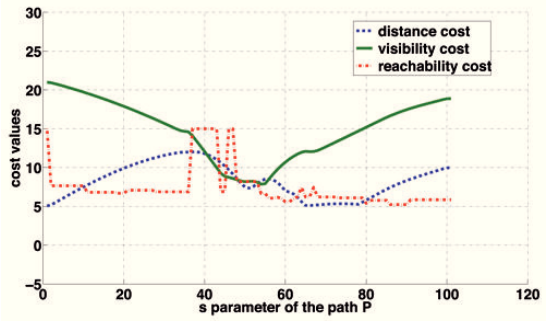
(c) Chemin initial, Contraintes mises à jour



(d) Profil de coût (c)



(e) Nouveau chemin, Contraintes mises à jour



(f) Profil de coût (e)

FIGURE 5.14 – Chemins et leurs profils de coût. Le chemin initial (a) maximise les critères de visibilité pour la position initiale de l'humain. Les nouvelles contraintes induites par le mouvement de l'humain (c) génèrent un profil de coût différent pour le chemin initial. Après la phase de replanification qui consiste à perturber de manière aléatoire le chemin (e) tient mieux compte de la sécurité de l'humain en augmentant les dégagements pour lui.

FIGURE 5.15 – Séquence complète pour donner l'objet

6

Conclusion et Perspectives

6.1 Conclusion

Les travaux présentés dans ce manuscrit portent sur la planification de mouvements en présence de contraintes d'interaction homme-robot. Les méthodes proposées permettent de tenir compte de la sécurité et du confort de l'homme mais également de l'efficacité avec laquelle sont réalisées les tâches coopératives.

Les méthodes développées sont basées sur des techniques d'échantillonnages aléatoires dont la performance a été démontrée pour de nombreuses applications. La contribution de ce travail repose sur l'extension de ces méthodes afin de tenir compte explicitement de la présence de l'homme dans des environnements contraints. L'efficacité des techniques mises en œuvre représente un premier pas vers la prise en compte de la dynamique introduite dans l'environnement par l'homme. Nous pouvons cependant noter que l'analyse des techniques globales et locales dépasse le cadre de l'interaction-homme robot et peut être appliquée à la planification de mouvements de bonne qualité en temps contraint.

Une première contribution de ce travail est de proposer un planificateur de mouvement de manipulation dédié à l'interaction homme-robot, intégrant la complémentarité de méthodes globales et locales. La planification est effectuée dans un espace de coût qui modélise des propriétés issues d'études anthropologiques sur le partage de l'espace et sur des études utilisateurs. La production d'un chemin initial est effectuée par une version bidirectionnelle de T-RRT qui est une variante de RRT qui applique des tests de transition afin de produire des chemins de bonne qualité. T-RRT explore l'espace cartésien afin d'accélérer la convergence. Le chemin est ensuite lissé tout en tenant compte de la carte de coût avant d'être traité par STOMP. Cette méthode

procède par déformation de la solution courante et produit des trajectoires lisses minimisant les accélérations le long de la trajectoire. La combinaison de cette technique de planification de mouvement avec la recherche du point d'échange permet de calculer des mouvements de transfert d'objet complet.

Une deuxième contribution concerne le partage de l'effort dans les stratégies d'échange d'objet du robot à l'homme. Le planificateur de transfert d'objet élaboré pour résoudre ce type de problème calcule des mouvements de navigation en combinant des techniques de grille et des techniques d'échantillonnage de manière efficace. Il permet de trouver rapidement des stratégies d'échange d'objets dans des environnements possiblement contraints en tenant compte de la sécurité, du confort mais également de la mobilité de l'humain. La méthode consiste à échantillonner des configurations d'échange et à évaluer leur faisabilité et leur coût qui intègre les différentes propriétés que nous venons d'énumérer. Cette approche est la première à notre connaissance à planifier les mouvements de l'homme. En fonction d'un paramètre que nous avons nommé mobilité, le planificateur retourne des stratégies d'échange plus ou moins rapide en fonction de l'effort requit de la part de l'humain.

Une étude utilisateur a été menée afin d'évaluer la pertinence de l'approche. Un ensemble de mesures objectives et subjectives sont analysées. Une tâche, assignée aux sujets, nous permet de moduler leur mobilité. Les stratégies planifiées avec une mobilité élevée sont plus efficaces pour réaliser la tâche et elle sont préférées par les sujets assignés à une tâche chronométrée.

Finalement, nous avons proposé une architecture logicielle complète pour effectuer des tâches de manipulation interactive. Elle présente également des contributions qui tiennent compte de la dynamique introduite par l'homme. Cette architecture a été évaluée sur le robot Jido du LAAS-CNRS sur un ensemble de sujets.

6.2 Améliorations et extensions

Amélioration algorithmique

Le T-RRT présenté dans le chapitre 3 a été étendu à une version bidirectionnelle qui permet de converger plus rapidement vers une solution de bonne qualité. L'extension de cette technique à un mode de fonctionnement "anytime" serait plus adaptée à la prise en compte de changements dans l'environnement. En effet une méthode de ce type améliore le chemin solution de manière monotone ce qui est souhaitable pour planifier un chemin dans une fenêtre de temps bornée. L'ajout de cycles dans l'arbre d'exploration ou l'utilisation de plusieurs arbres de recherche permettrait d'obtenir ce mode de fonctionnement. Dans ce cas l'exploration plus globale de la carte de coût permettrait une convergence vers le chemin optimal de manière similaire à RRT*. Une autre amélioration possible serait l'adaptation de la puissance du filtrage afin d'obtenir rapidement une solution qui puisse être optimisée par la suite si il reste du temps.

Evaluation de la mobilité et replanification

La mobilité qui est prise en compte par le planificateur de partage d'effort du chapitre 4 dépend de plusieurs paramètres : la tâche, les capacités physiques du receveur ou l'urgence de d'obtenir l'objet. L'évaluation de la mobilité est difficile mais essentielle pour le fonctionnement du planificateur. Elle doit intégrer une combinaison de critères qui peuvent être définis par l'utilisateur lui-même ou inféré par le robot.

En plus du problème de l'estimation de la mobilité, le mouvement planifié pour l'humain peut être différent de celui observé pendant l'exécution. L'homme peut se déplacer de manière contraire ou rester immobile alors que le plan produit par le robot exige un déplacement. Dans ces deux cas, si la tâche n'est pas explicitement annulée, il est nécessaire de replanifier une configuration de transfert. L'ajustement de la mobilité et donc son évaluation dynamique joue également un rôle important dans ce cas.

Réutilisation de plan pour la manipulation

Les planificateurs de tâches de manipulation ne tiennent généralement pas compte des planifications passées. STOMP présenté dans le chapitre 3, permet de déformer un chemin solution. Il est donc possible de réutiliser un chemin d'une planification antérieure dans l'esprit de [Berenson 12]. Le robot peut ainsi apprendre par expérience. Le choix du plan initial est alors déterminant. Ce type d'approche peut-être appliqué au mouvement en interaction avec l'homme ou à des tâches de manipulation simple.

Tâches de manipulation interactive plus réactives

Les planificateurs que nous avons développés dans 3 produisent des chemins sans tenir compte de leur exécution. Le mode de contrôle est important notamment dans les tâches d'échange d'objet ou de saisie pour lesquelles une boucle de rétrocontrôle est généralement utilisée avec un capteur externe (e.g. caméra, capteur de profondeur). Dans ces cas les trajectoires produites par le planificateur ne sont pas exactement celles exécutées par le robot. Il peut être important de prendre en compte des contraintes qui permettent au contrôleur d'exécuter le chemin planifié comme le dégagement aux obstacles ou un écartement des butées articulaires.

Généricité des techniques de planification

Les techniques développées dans cette thèse s'intègrent dans un effort plus large sur l'autonomie des systèmes robotiques. En effet la génération de mouvements sûrs, confortables et lisibles est une capacité essentielle pour l'aboutissement de la robotique de service. De plus la modélisation de l'humain et de son comportement doit être prise en compte à tous les niveaux de l'architecture d'un tel robot. Cependant les techniques utilisées pour calculer des chemins de bonne qualité sont génériques et leur application ne se limite pas à la planification de mouvement en interaction avec l'homme. La prise en compte d'un coût général permet de générer des mouvements prenant en compte un large ensemble de contraintes. Ces techniques ont été appliquées à la

prise en compte de l'incertitude [Berenson 11] et à des problèmes de biologie moléculaire [Iehl 12]. Ces méthodes de planification peuvent être étendues pour prendre en compte des systèmes divers avec des contraintes dynamiques.

Bibliographie

[Alami 06] R Alami & et al. *Safe and Dependable Physical Human-Robot Interaction in Anthropic Domains : State of the Art and Challenges.* In Procceedings IROS Workshop on pHRI, Beijing, China, 2006. 18

[Althaus 04] P. Althaus, H. Ishiguro, T. Kanda, T. Miyashita & H.I. Christensen. *Navigation for human-robot interaction tasks.* In Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on, volume 2, pages 1894–1900. IEEE, 2004. 21, 26

[Amato 98] N.M. Amato, O.B. Bayazit, L.K. Dale, C. Jones & D. Vallejo. *OBPRM : an obstacle-based PRM for 3D workspaces.* pages 155–168. A. K. Peters, Ltd., Natick, MA, USA, 1998. 9

[Amato 00] N.M. Amato, O.B. Bayazit, L.K. Dale, C. Jones & D. Vallejo. *Choosing good distance metrics and local planners for probabilistic roadmap methods.* vol. 16, no. 4, pages 442–447, August 2000. 10

[Arbib 98] Michael A. Arbib. *Schema theory.* In The handbook of brain theory and neural networks, pages 830–834. MIT Press, Cambridge, MA, USA, 1998. 100

[Astrom 70] K.J. Astrom. Introduction to stochastic control theory. Dover Publications, 1970. 36

[Baginski 97] B. Baginski. *Efficient dynamic collision detection using expanded geometry models.* In Intelligent Robots and Systems, 1997. IROS'97., Proceedings of the 1997 IEEE/RSJ International Conference on, volume 3, pages 1714–1720. IEEE, 1997. 33

[Barraquand 91] J. Barraquand & J.-C. Latombe. *Robot Motion Planning : A Distributed Representation Approach.* International Journal Of Robotic Reasearch, vol. 10, no. 6, pages 628–649, December 1991. 7, 11

[Bekris 07] K.E. Bekris & L.E. Kavraki. *Greedy but safe replanning under kinodynamic constraints.* In Robotics and Automation, 2007 IEEE International Conference on, pages 704–710. IEEE, 2007. 14

[Bennewitz 05] M. Bennewitz, W. Burgard, G. Cielniak & S. Thrun. *Learning motion patterns of people for compliant robot motion.* The International Journal of Robotics Research, vol. 24, no. 1, pages 31–48, 2005. 21

[Berchtold 94] S. Berchtold & B. Glavina. *A scalable optimizer for automatically generated manipulator motions.* In IEEE/RSJ Int. Conf. on Intel. Rob. And Sys., 1994. 16, 31, 33, 47

[Berenson 09] D. Berenson, S.S. Srinivasa, D. Ferguson & J.J. Kuffner. *Manipulation planning on constraint manifolds.* In Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, pages 625–632. IEEE, 2009. 13

[Berenson 11] D. Berenson, T. Siméon & S.S. Srinivasa. *Addressing Cost-Space Chasms in Manipulation Planning.* In IEEE Int. Conf. Robot. And Autom., 2011. 116

[Berenson 12] D. Berenson, P. Abbeel & K. Goldberg. *A robot path planning framework that learns from experience.* In Robotics and Automation (ICRA), 2012 IEEE International Conference on, pages 3671–3678. IEEE, 2012. 115

[Bertsekas 78] D.P. Bertsekas & S.E. Shreve. Stochastic optimal control : The discrete time case, volume 139. Academic Press NY, 1978. 36

[Bicchi 04] A. Bicchi & G. Tonietti. *Fast and Soft Arm Tactics : Dealing with the Safety-Performance Trade-Off in Robot Arms Design and Control.* Robotics and Automation Magazine, 2004. 18

[Boekhorst 05] R. Boekhorst, M.L. Walters, K.L. Koay, K. Dautenhahn & C.L. Nehaniv. *A study of a single robot interacting with groups of children in a rotation game scenario.* In Computational Intelligence in Robotics and Automation, 2005. CIRA 2005. Proceedings. 2005 IEEE International Symposium on, pages 35–40. IEEE, 2005. 20

[Bohlin 00] R. Bohlin & L.E. Kavraki. *Path planning using lazy PRM.* In IEEE Int. Conf. Robot. And Autom., volume 1, pages 521–528, 2000. 10, 11

[Boor 99] V. Boor, M.H. Overmars & A.F. Van Der Stappen. *The Gaussian sampling strategy for probabilistic roadmap planners.* In IEEE Int. Conf. Robot. And Autom., volume 2, pages 1018–1023, 1999. 10

[Bounab 10] Belkacem Bounab, Abdenour Labed & Daniel Sidobre. *Stochastic optimization-based approach for multifingered grasps synthesis.* Robotica, vol. 28, no. 07, pages 1021–1032, 2010. 92

[Breazeal 02] Cynthia Breazeal. Designing sociable robots. MIT Press, Cambridge, MA, USA, 2002. 99

[Breazeal 05] Cynthia Breazeal, Cory D. Kidd, Andrea Lockerd Thomaz, Guy Hoffman & Matt Berlin. *Effects of nonverbal communication on efficiency and robustness in human-robot teamwork.* In in IROS-2005, pages 383–388, Edmonton, Alberta, Canada, 2005. ACM/IEEE. 99

[Brock 00] O. Brock & O. Khatib. *Real-time re-planning in high-dimensional configuration spaces using sets of homotopic paths.* In IEEE Int. Conf. Robot. And Autom., 2000. 90

[Brock 02] O. Brock & O. Khatib. *Elastic strips : A framework for motion generation in human environments.* The International Journal of Robotics Research, vol. 21, no. 12, pages 1031–1052, 2002. 17

[Broquère 08] X. Broquère, D. Sidobre & I. Herrera-Aguilar. *Soft motion trajectory planner for service manipulator robot.* In IEEE/RSJ Int. Conf. on Intel. Rob. And Sys (IROS), 2008. 78, 91, 94, 98

[Broquère 10] X. Broquère & D. Sidobre. *From motion planning to trajectory control with bounded jerk for service manipulator robots.* In IEEE Int. Conf. Robot. And Autom., 2010. 91, 94, 96

[Broquère 11] X. Broquère. *Planification de trajectoire pour la manipulation d'objets et l'interaction Homme-robot.* PhD thesis, Université Paul Sabatier-Toulouse III, 2011. 50, 94, 104

[Burattini 08] E. Burattini & S. Rossi. *Periodic Adaptive Activation of Behaviors in Robotic System.* IJPRAI - Special Issue on Brain, Vision and Artificial Intelligence, vol. 22, no. 5, pages 987–999, 2008. 100

[Burattini 10] Ernesto Burattini, Silvia Rossi, Alberto Finzi & Mariacarla Staffa. *Attentional Modulation of Mutually Dependent Behaviors.* In Proc. of SAB-2010, pages 283–292, Paris, France, 2010. LNAI 6226. 100

[Cakmak 11] M. Cakmak, S.S. Srinivasa, M.K. Lee, S. Kiesler & J. Forlizzi. *Using spatial and temporal contrast for fluent robot-human hand-overs.* ACM, 2011. 27, 28

[Canny 88] J.F. Canny. The complexity of robot motion planning. MIT Press, Cambridge, MA, USA, 1988. 7

[Chang 95] H. Chang & T.-Y. Li. *Assembly maintainability study with motion planning.* In IEEE Int. Conf. Robot. And Autom., volume 1, pages 1012–1019, may 1995. 11

[Chen 98] P.C. Chen & Y.K. Hwang. *SANDROS : a dynamic graph search algorithm for motion planning.* Robotics and Automation, IEEE Transactions on, vol. 14, no. 3, pages 390–403, 1998. 33, 47

[Choset 05a] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki & S. Thrun. Principles of robot motion : Theory, algorithms, and implementations (intelligent robotics and autonomous agents). The MIT Press, June 2005. 5, 7

[Choset 05b] H.M. Choset, S. Hutchinson, K.M. Lynch, G. Kantor, W. Burgard, L.E. Kavraki & S. Thrun. Principles of robot motion : theory, algorithms, and implementation. The MIT Press, 2005. 29, 65

[Cooper 00] Richard Cooper & Tim Shallice. *Contention scheduling and the control of routine activities.* Cognitive Neuropsychology, vol. 17, pages 297–338, 2000. 99

[Cortes 02] J. Cortes, T. Simeon & J.P. Laumond. *A random loop generator for planning the motions of closed kinematic chains using PRM methods.* In Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on, volume 2, pages 2141–2146. IEEE, 2002. 44

[Cortés 05] J. Cortés & T. Siméon. *Sampling-based motion planning under kinematic loop-closure constraints.* Algorithmic Foundations of Robotics VI, pages 75–90, 2005. 26, 43

[Cortés 08] J. Cortés, L. Jaillet & T. Siméon. *Disassembly Path Planning for Complex Articulated Objects.* vol. 24, no. 2, pages 475–481, april 2008. 13

[Dalibard 09] S. Dalibard & J.P. Laumond. *Control of probabilistic diffusion in motion planning.* Algorithmic Foundation of Robotics VIII, pages 467–481, 2009. 13

[Dautenhahn 06] K. Dautenhahn, M. Walters, S. Woods, K. L. Koay, C. L. Nehaniv, E. A. Sisbot, R. Alami & T. Siméon. *How may I serve you ? : a robot companion approaching a seated person in a helping context.* In ACM SIGCHI/SIGART International Conference on Human-Robot Interaction, HRI, pages 172–179, Utah, USA, March 2006. 19

[Dayan 97] P. Dayan & G.E. Hinton. *Using expectation-maximization for reinforcement learning.* Neural Computation, vol. 9, no. 2, pages 271–278, 1997. 35

[Dehais 11] F. Dehais, E.A. Sisbot, R. Alami & M. Causse. *Physiological and subjective evaluation of a human-robot object hand-over task.* Applied Ergonomics, 2011. 28

[Delsart 08] V. Delsart & T. Fraichard. *Navigating dynamic environments using trajectory deformation.* In Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on, pages 226–233. IEEE, 2008. 17

[Dolgov 09] D. Dolgov, S. Thrun, M. Montemerlo & J. Diebel. *Path planning for autonomous driving in unknown environments.* In Experimental Robotics, pages 55–64. Springer, 2009. 14

[Edsinger 07] A. Edsinger & C. C. Kemp. *Human-Robot Interaction for Cooperative Manipulation : Handing Objects to One Another.* In RO-MAN 2007, pages 1167–1172, Jeju, Korea, 2007. IEEE. 99, 108

[Edsinger 08] A. Edsinger & C.C. Kemp. *Human-robot interaction for cooperative manipulation : Handing objects to one another.* In RO-MAN 2007. IEEE, 2008. 26, 27

[Faverjon 84] B. Faverjon. *Obstacle avoidance using an octree in the configuration space of a manipulator*. In IEEE Int. Conf. Robot. And Autom., volume 1, pages 504–512, march 1984. 7

[Felzenszwalb 04] P. Felzenszwalb & D. Huttenlocher. *Distance transforms of sampled functions*. 2004. 49

[Ferguson 06a] D. Ferguson, N. Kalra & A. Stentz. *Replanning with rrts*. In Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on, pages 1243–1248. IEEE, 2006. 14

[Ferguson 06b] D. Ferguson & A. Stentz. *Anytime rrts*. In IEEE/RSJ Int. Conf. on Intel. Rob. And Sys., 2006. 15, 90

[Flash 85] T. Flash & N. Hogan. *The coordination of arm movements : an experimentally confirmed mathematical model*. Journal of neuroscience, 1985. 27, 90

[Fleury 97] S. Fleury, M. Herrb & R. Chatila. *Genom : A tool for the specification and the implementation of operating modules in a distributed robot architecture*. In IEEE/RSJ Int. Conf. on Intel. Rob. And Sys., 1997. 104

[Geraerts 07] R. Geraerts & M.H. Overmars. *Creating high-quality paths for motion planning*. The International Journal of Robotics Research, 2007. 16

[Goodrich 07] M.A. Goodrich & A.C. Schultz. *Human-robot interaction : a survey*. Foundations and Trends in Human-Computer Interaction, vol. 1, no. 3, pages 203–275, 2007. 18

[Guernane 09] R. Guernane & N. Achour. *An Algorithm for Generating Safe and Execution-Optimized Paths*. pages 16–21, april 2009. 17

[Haddadin 07] S. Haddadin, A. Albu-Schäffer & G. Hirzinger. *Safety evaluation of physical human-robot interaction via crash-testing*. In Robotics : science and systems conference (RSS2007), pages 217–224, 2007. 20

[Haddadin 08] S. Haddadin, A. Albu-Schaffer & G. Hirzinger. *The role of the robot mass and velocity in physical human-robot interaction-part i : Non-constrained blunt impacts*. In Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, pages 1331–1338. IEEE, 2008. 20

[Hall 63] Edward T. Hall. *A system for the notation of proxemic behavior*. American anthropologist, 1963. 29, 38, 66, 68

[Hall 66] E. T. Hall. The hidden dimension. Doubleday, Garden City, N.Y., 1966. 2, 18, 21, 24

[Hoeller 07] F. Hoeller, D. Schulz, M. Moors & F.E. Schneider. *Accompanying persons with a mobile robot using motion prediction and probabilistic roadmaps*. In Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on, pages 1260–1265. IEEE, 2007. 21, 25

[Hsu 97] D. Hsu, J.C. Latombe & R. Motwani. *Path planning in expansive configuration spaces.* In Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on, volume 3, pages 2719–2726. IEEE, 1997. 9, 12, 25

[Hsu 98] D. Hsu, L.E. Kavraki, J.C. Latombe, R. Motwani & S. Sorkin. *On finding narrow passages with probabilistic roadmap planners.* In P.K. Agarwal *et al*, editeurs, wafr, pages 141–154. A. K. Peters, Wellesley, MA, 1998. 9

[Hsu 00] D. Hsu. *Randomized Single-query Motion Planning in Expansive Spaces.* PhD thesis, Deptartement of Computer Science, Stanford University, 2000. 16

[Hsu 03] D. Hsu, T. Jiang, J. Reif & Z. Sun. *The bridge test for sampling narrow passages with probabilistic roadmap planners.* In IEEE Int. Conf. Robot. And Autom., volume 3, pages 4420–4426, sept. 2003. 10

[Huber 08] M. Huber, M. Rickert, A. Knoll, T. Brandt & S. Glasauer. *Human-robot interaction in handing-over tasks.* In Robot and Human Interactive Communication, 2008. RO-MAN 2008. The 17th IEEE International Symposium on, pages 107–112. IEEE, 2008. 20, 27, 91

[Iehl 12] R. Iehl, J. Cortes & T. Simeon. *Costmap planning in high dimensional configuration spaces.* In Advanced Intelligent Mechatronics (AIM), 2012 IEEE/ASME International Conference on, pages 166–172. IEEE, 2012. 16, 116

[Jaillet 04] L. Jaillet & T. Siméon. *A PRM-based motion planner for dynamically changing environments.* In Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on, volume 2, pages 1606–1611. IEEE, 2004. 14

[Jaillet 05] L. Jaillet, A. Yershova, S.M. LaValle & T. Siméon. *Adaptive tuning of the sampling domain for dynamic-domain RRTs.* In IEEE/RSJ Int. Conf. on Intel. Rob. And Sys., pages 2851–2856, August 2005. 13

[Jaillet 08a] L. Jaillet & T. Siméon. *Path Deformation Roadmaps : Compact Graphs with Useful Cycles for Motion Planning.* International Journal Of Robotic Reasearch, vol. 27, no. 11–12, pages 1175–1188, 2008. 10

[Jaillet 08b] L. Jaillet & T. Siméon. *Path deformation roadmaps : Compact graphs with useful cycles for motion planning.* The International Journal of Robotics Research, 2008. 90

[Jaillet 10] L. Jaillet, J. Cortés & T. Siméon. *Sampling-based path planning on configuration-space costmaps.* IEEE Transactions on Robotics, 2010. 15, 30, 31, 32, 45, 57, 58

[Jiménez 98] P. Jiménez, F. Thomas & C. Torras. *Collision Detection Algorithms for Motion Planning. Robot Motion Planning and Control.* In J.-P. Laumond,

editeur, Lecture Notes in Control and Information Sciences, 229, pages 305–343. Springer, 1998. 11

[Kahneman 73] D. Kahneman. Attention and effort. Prentice-Hall, Englewood Cliffs, NJ, 1973. 99, 100

[Kajikawa 95] S. Kajikawa, T. Okino, K. Ohba & H. Inooka. *Motion planning for hand-over between human and robot.* In iros, page 193. Published by the IEEE Computer Society, 1995. 27

[Kalakrishnan 11] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor & S. Schaal. *STOMP : Stochastic Trajectory Optimization for Motion Planning.* In icra, 2011. 17, 30

[Kallman 04] M. Kallman & M. Mataric. *Motion planning using dynamic roadmaps.* In Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on, volume 5, pages 4399–4404. IEEE, 2004. 14

[Kaplan 06] Frederic Kaplan & Verena V. Hafner. *The challenges of joint attention.* Interaction Studies, vol. 7, no. 2, pages 135–169, 2006. 99

[Karaman 11] S. Karaman & E. Frazzoli. *Sampling-based algorithms for optimal motion planning.* International Journal of Robotics Research, vol. 30, no. 7, pages 846–894, 2011. 11, 15, 16, 57

[Kavraki 95] L.E. Kavraki. *Random Networks in Configuration Space for Fast Path Planning.* PhD thesis, Stanford University, Stanford, CA, USA, 1995. 9

[Kavraki 96] L.E. Kavraki, P. Svestka, J.-C. Latombe & M.H. Overmars. *Probabilistic roadmaps for path planning in high-dimensional configuration spaces.* vol. 12, no. 4, pages 566 –580, aug 1996. 1, 8, 9, 10

[Kavraki 98] L.E. Kavraki & J.C. Latombe. *Probabilistic roadmaps for robot path planning.* 1998. 11, 16, 33, 47

[Kemp 07] C.C. Kemp, A. Edsinger & E. Torres-Jara. *Challenges for robot manipulation in human environments [grand challenges of robotics].* Robotics & Automation Magazine, IEEE, vol. 14, no. 1, pages 20–29, 2007. 18

[Kendon 10] A. Kendon. *Spacing and orientation in co-present interaction.* Development of Multimodal Interfaces : Active Listening and Synchrony, pages 1–15, 2010. 24

[Khanmohammadi 08] S. Khanmohammadi & A. Mahdizadeh. *Density Avoided Sampling : An Intelligent Sampling Technique for Rapidly-Exploring Random Trees.* pages 672–677, sept. 2008. 13

[Khatib 86] O Khatib. *Real-time obstacle avoidance for manipulators and mobile robots.* International Journal Of Robotic Reasearch, vol. 5, no. 1, pages 90–98, 1986. 7

[Koay 07] K. L. Koay, E. Akin Sisbot, D. A. Syrdal, M. L. Walters, K. Dautenhahn & R. Alami. *Exploratory Study of a Robot Approaching a Person in the Context of Handling Over an Object.* In AAAI, Palo Alto, CA, USA, 2007. 19, 20, 29, 68

[Koenig 02] S. Koenig & M. Likhachev. *D^* Lite.* In Proceedings of the national conference on artificial intelligence, pages 476–483. Menlo Park, CA ; Cambridge, MA ; London ; AAAI Press ; MIT Press ; 1999, 2002. 14

[Koren 91] Y. Koren & J. Borenstein. *Potential field methods and their inherent limitations for mobile robot navigation.* In Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on, pages 1398–1404. IEEE, 1991. 7

[Kruse 12] T. Kruse, P. Basili, S. Glasauer & A. Kirsch. *Legible robot navigation in the proximity of moving humans.* In Advanced Robotics and its Social Impacts (ARSO), 2012 IEEE Workshop on, pages 83–88. IEEE, 2012. 23

[Kuffner 00] Jr. Kuffner J.J. & S.M. LaValle. *RRT-connect : An efficient approach to single-query path planning.* In IEEE Int. Conf. Robot. And Autom., volume 2, pages 995–1001, 2000. 12, 13

[Kulić 05] D. Kulić & E.A. Croft. *Safe planning for human-robot interaction.* Journal of Robotic Systems, vol. 22, no. 7, pages 383–396, 2005. 21

[Kulic 07a] D. Kulic & E. Croft. *Physiological and subjective responses to articulated robot motion.* Robotica, vol. 25, pages 13–27, 2007. 27

[Kulic 07b] D. Kulic & E. Croft. *Physiological and subjective responses to articulated robot motion.* Robotica, 2007. 66

[Kuwata 09] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, JP How & G. Fiore. *Real-time motion planning with applications to autonomous urban driving.* Control Systems Technology, IEEE Transactions on, vol. 17, no. 5, pages 1105–1118, 2009. 14

[Lam 10] C.P. Lam, C.T. Chou, K.H. Chiang & L.C. Fu. *Human-Centered Robot Navigation, Towards a Harmoniously Human-Robot Coexisting Environment.* Robotics, IEEE Transactions on, 2010. 66

[Lam 11] C.P. Lam, C.T. Chou, K.H. Chiang & L.C. Fu. *Human-Centered Robot Navigation - Towards a Harmoniously Human - Robot Coexisting Environment.* Robotics, IEEE Transactions on, vol. 27, no. 1, pages 99–112, 2011. 21, 23

[Lamiraux 04] F. Lamiraux, D. Bonnafous & O. Lefebvre. *Reactive path deformation for nonholonomic mobile robots.* Robotics, IEEE Transactions on, vol. 20, no. 6, pages 967–977, 2004. 17

[Lang 03] Sebastian Lang, Marcus Kleinehagenbrock, Sascha Hohenner, Jannik Fritsch, Gernot A. Fink & Gerhard Sagerer. *Providing the Basis for*

Human-Robot-Interaction : A Multi-Modal Attention System for a Mobile Robot. In Proc. Int. Conf. on Multimodal Interfaces, pages 28–35, Vancouver, Canada, 2003. ACM. 99

[Larsen 00] E. Larsen, S. Gottschalk, M.C. Lin & D. Manocha. *Fast distance queries with rectangular swept sphere volumes.* In IEEE Int. Conf. Robot. And Autom., volume 4, pages 3719–3726, 2000. 11

[Latombe 91a] J.-C. Latombe. Robot motion planning. Kluwer Academic Publishers, Boston, MA, 1991. 1, 5, 7

[Latombe 91b] J.C. Latombe. Robot motion planning. Springer, 1991. 65, 71

[Laumond 98] J. P. Laumond, S. Sekhavat & F. Lamiraux. *Guidelines in nonholonomic motion planning for mobile robots.* In J.-P. Laumond, editeur, Robot Motion Plannning and Control, volume 229/1998, pages 1–53. Springer-Verlag, 1998. 10

[LaValle 98] S.M. LaValle. *Rapidly-exploring random trees : A new tool for path planning.* Rapport technique 98–11, Computer Science Departement, Iowa State University, October 1998. 8, 9, 12

[LaValle 01a] S. M. LaValle & J. Kuffner. *Rapidly-exploring random trees : Progress and prospects.* In Workshop on the Algorithmic Foundations of Robotics, 2001. 14

[LaValle 01b] Steven M. LaValle & J. Kuffner. *Rapidly-Exploring Random Trees : Progress and Prospects.* In Algorithmic and Computational Robotics : New Directions, pages 293–308, Wellesley, MA, 2001. 12

[LaValle 02] S.M. LaValle. *From Dynamic Programming to RRTs : Algorithmic Design of Feasible Trajectories.* In Springer Berlin /Heidelberg, editeur, Control Problems in Robotics, volume 4/2003, pages 19–37. Springer-Verlag, 2002. 10

[LaValle 04] S.M. LaValle, M.S. Branicky & S.R. Lindemann. *On the Relationship between Classical Grid Search and Probabilistic Roadmaps.* International Journal Of Robotic Reasearch, vol. 23, pages 673–692, 2004. 10

[LaValle 06] S.M. LaValle. Planning algorithms. Cambridge Univ Pr, 2006. 1, 5, 7, 29, 65

[Lawrence-Zúniga 03] D. Lawrence-Zúniga. The anthropology of space and place : Locating culture, volume 4. Wiley-Blackwell, 2003. 18

[Lawrence 94] C.T. Lawrence, J.L. Zhou & A.L. Tits. *User's Guide for CFSQP Version 2.0 : AC Code for Solving (Large Scale) Constrained Nonlinear (Minimax) Optimization Problems, Generating Iterates Satisfying All Inequality Constraints.* 1994. 17

[Le 09] D.T. Le, J. Cortés & T. Siméon. *A path planning approach to (dis)assembly sequencing.* pages 286–291, aug. 2009. 13

[Leven 00] P. Leven & S. Hutchinson. *Toward real-time path planning in changing environments.* 2000. 14

[Likhachev 03] M. Likhachev, G. Gordon & S. Thrun. *ARA* : Anytime A* with provable bounds on sub-optimality.* Advances in Neural Information Processing Systems (NIPS), vol. 16, 2003. 14

[Likhachev 08] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz & S. Thrun. *Anytime search in dynamic graphs.* Artificial Intelligence, vol. 172, no. 14, pages 1613–1643, 2008. 14

[Likhachev 09] M. Likhachev & D. Ferguson. *Planning long dynamically feasible maneuvers for autonomous vehicles.* The International Journal of Robotics Research, vol. 28, no. 8, pages 933–945, 2009. 14

[Lin 03] M.C. Lin & D. Manocha. *Collision and Proximity Queries*, 2003. 11

[Littlejohn 07] S.W. Littlejohn & K.A. Foss. Theories of human communication. Wadsworth Pub Co, 2007. 18

[Lozano-Pérez 83] T. Lozano-Pérez. *Spatial Planning : A Configuration Space Approach.* IEEE Transactions on Computers, vol. C–32, no. 2, pages 108–120, Feb. 1983. 5, 6

[Lozano-Pérez 87] T. Lozano-Pérez. *A simple motion-planning algorithm for general robot manipulators.* vol. 3, no. 3, pages 224–238, june 1987. 7

[Madhava Krishna 06] K. Madhava Krishna, R. Alami & T. Simeon. *Safe proactive plans and their execution.* Robotics and Autonomous Systems, vol. 54, no. 3, pages 244–255, 2006. 22

[Marler 05] R.T. Marler, S. Rahmatalla, M. Shanahan & K. Abdel-Malek. *A new discomfort function for optimization-based posture prediction.* 2005. 39, 68

[Martinez-Garcia 05] E.A. Martinez-Garcia, O. Akihisa *et al. Crowding and guiding groups of humans by teams of mobile robots.* In Advanced Robotics and its Social Impacts, 2005. IEEE Workshop on, pages 91–96. IEEE, 2005. 21, 26

[Moulard 12] T. Moulard. *Optimisation numérique pour la robotique et exécution de trajectoires référencées capteurs.* PhD thesis, Institut National Polytechnique de Toulouse-INPT, 2012. 17

[Nagai 03] Yukie Nagai, Koh Hosoda, Akio Morita & Minoru Asada. *A constructive model for the development of joint attention.* Connect. Sci., vol. 15, no. 4, pages 211–229, 2003. 99

[Nakamura 87] Y. Nakamura & H. Hanafusa. *Optimal redundancy control of robot manipulators.* The International journal of robotics research, vol. 6, no. 1, pages 32–42, 1987. 26

[Nakauchi 02] Y. Nakauchi & R. Simmons. *A social robot that stands in line.* Autonomous Robots, vol. 12, no. 3, pages 313–324, 2002. 21, 26

[Nielsen 00] C.L. Nielsen & L.E. Kavraki. *A two level fuzzy PRM for manipulation planning.* In IEEE/RSJ Int. Conf. on Intel. Rob. And Sys., volume 3, pages 1716–1721, 2000. 11

[Nieuwenhuisen 04] D. Nieuwenhuisen & M.H. Overmars. *Useful cycles in probabilistic roadmap graphs.* In IEEE Int. Conf. Robot. And Autom., volume 1, pages 446–452, april 2004. 10

[Nilsson 69] N.J. Nilsson. *A mobius automation : an application of artificial intelligence techniques.* volume 1, pages 509–520, San Francisco, CA, USA, 1969. Morgan Kaufmann Publishers Inc. 7

[Nissoux 99] C. Nissoux. *Visibilité et méthodes probabilistes pour la planification de mouvements en robotique.* PhD thesis, Université de Toulouse 3, Toulouse, FRANCE, 1999. 10

[Nonaka 04] S. Nonaka, K. Inoue, T. Arai & Y. Mae. *Evaluation of human sense of security for coexisting robots using virtual reality. 1st report : evaluation of pick and place motion of humanoid robots.* In IEEE Int. Conf. Robot. And Autom., New Orleans, USA, 2004. 18

[Norman 86] DA Norman & T. Shallice. *Attention in action : willed and automatic control of behaviour.* Consciousness and Self-regulation : advances in research and theory, vol. 4, pages 1–18, 1986. 99

[Oriolo 09] G. Oriolo & M. Vendittelli. *A control-based approach to task-constrained motion planning.* In Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, pages 297–302. IEEE, 2009. 13

[Ott 06] C. Ott. *A Humanoid Two-Arm System for Dexterous Manipulation.* In IEEE-RAS Int. Conf. on Hum. Robot., 2006. 51, 97

[Pacchierotti 05] E. Pacchierotti, H.I. Christensen & P. Jensfelt. *Human-robot embodied interaction in hallway settings : a pilot user study.* In IEEE International Workshop on Robot and Human Interactive Communication, RO-MAN, pages 164–171, Nashville, USA, August 2005. 19, 21

[Pacchierotti 06a] E. Pacchierotti, H. Christensen & P. Jensfelt. *Embodied social interaction for service robots in hallway environments.* In Field and Service Robotics, pages 293–304. Springer, 2006. 25

[Pacchierotti 06b] E. Pacchierotti, H.I. Christensen & P. Jensfelt. *Evaluation of Passing Distance for Social Robots.* In IEEE International Workshop on Robot and Human Interactive Communication, RO-MAN, pages 315–320, Hatfield, UK, September 2006. 19

[Pandey 09] A.K. Pandey & R. Alami. *A step towards a sociable robot guide which monitors and adapts to the person's activities.* In Advanced Robotics,

2009. ICAR 2009. International Conference on, pages 1–8. IEEE, 2009. 21

[Petti 05] S. Petti & T. Fraichard. *Safe motion planning in dynamic environments.* In Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on, pages 2210–2215. IEEE, 2005. 14

[Posner 75] M.I. Posner & C.R.R. Snyder. *Attention and cognitive control.* In Information processing and cognition : The Loyola Symposium, Hillsdale, NJ : Erlbaum, 1975. Psychology Pr. 99

[Prince 02] S. Prince, A.D. Cheok, F. Farbiz, T. Williamson, N. Johnson, M. Billinghurst & H. Kato. *3d live : Real time captured content for mixed reality.* In ISMAR 2002, 2002. 78, 104

[Quigley 09] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler & A. Ng. *ROS : an open-source Robot Operating System.* In ICRA workshop on open source software, volume 3, 2009. 78

[Quinlan 93] S. Quinlan & O. Khatib. *Elastic bands : Connecting path planning and control.* 1993. 16, 17, 34

[Ratliff 09] N. Ratliff, M. Zucker, J.A. Bagnell & S. Srinivasa. *Chomp : Gradient optimization techniques for efficient motion planning.* In Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, pages 489–494. IEEE, 2009. 17, 35

[Reif 79] J.H. Reif. *Complexity of the mover's problem and generalizations.* In Symposium on Foundations of Computer Science, pages 421–427, Washington, DC, USA, 1979. IEEE Computer Society. 7

[Rios-Martinez 12] J. Rios-Martinez, A. Renzaglia, A. Spalanzani, A. Martinelli & C. Laugier. *Navigating between people : a stochastic optimization approach.* In Robotics and Automation (ICRA), 2012 IEEE International Conference on, pages 2880–2885. IEEE, 2012. 24

[Robert 99] C.P. Robert, G. Casella & C.P. Robert. Monte carlo statistical methods, volume 2. Springer New York, 1999. 31

[Rodriguez 06] S. Rodriguez, X. Tang, J.-M. Lien & N.M. Amato. *An obstacle-based rapidly-exploring random tree.* In IEEE Int. Conf. Robot. And Autom., pages 895–900, may 2006. 13

[Sakata 04] K. Sakata, T. Takubo, K. Inoue, S. Nonaka, Y. Mae & T. Arai. *Psychological evaluation on shape and motions of real humanoid robot.* In Robot and Human Interactive Communication, 2004. ROMAN 2004. 13th IEEE International Workshop on, pages 29–34. IEEE, 2004. 20

[Sasaki 06] T. Sasaki & H. Hashimoto. *Human observation based mobile robot navigation in intelligent space.* In Intelligent Robots and Systems, 2006

IEEE/RSJ International Conference on, pages 1044–1049. IEEE, 2006. 22

[Saut 12] Jean-Philippe Saut & Daniel Sidobre. *Efficient models for grasp planning with a multi-fingered hand.* Robotics and Autonomous Systems, vol. 60, no. 3, pages 347 – 357, 2012. Autonomous Grasping. 92

[Scandolo 11a] L. Scandolo & T. Fraichard. *An Anthropomorphic Navigation Scheme for Dynamic Scenarios.* In IEEE Int. Conf. Robot. And Autom. (ICRA), 2011. 66

[Scandolo 11b] L. Scandolo & T. Fraichard. *An anthropomorphic navigation scheme for dynamic scenarios.* In Robotics and Automation (ICRA), 2011 IEEE International Conference on, pages 809–814. IEEE, 2011. 23

[Schwartz 83] J. T. Schwartz & M. Sharir. *On the Piano Movers' Problem : III. Coordinating the Motion of Several Independent Bodies.* International Journal Of Robotic Reasearch, vol. 2, pages 46–75, September 1983. 6, 7

[Senders 64] J. Senders. *The human operator as a monitor and controller of multidegree of freedom systems.* IEEE Trans. on Human Factors in Electronics, HFE-5, pages 2–6, 1964. 100

[Shah 11] J. Shah, J. Wiken, B. Williams & C. Breazeal. *Improved human-robot team performance using chaski, a human-inspired plan execution system.* In Proceedings of the 6th international conference on Human-robot interaction, pages 29–36. ACM, 2011. 65

[Shibata 95] S. Shibata, K. Tanaka & A. Shimizu. *Experimental analysis of handing over.* In Robot and Human Communication, RO-MAN, Proceedings., 4th IEEE International Workshop on, 1995. 27

[Simeon 01] T. Simeon, JP Laumond & F. Lamiraux. *Move3D : a generic platform for path planning.* In 4th Int. Symp. on Assembly and Task Planning. Citeseer, 2001. 51, 75

[Sisbot 07a] E. A. Sisbot, L. F. Marin-Urias, R. Alami & T. Siméon. *Human Aware Mobile Robot Motion Planner.* IEEE Transactions on Robotics, 2007. 21, 23, 24, 29, 38, 40, 42, 66, 68, 91

[Sisbot 07b] E. A. Sisbot, L. F. Marin-Urias, R. Alami & T. Siméon. *Spatial Reasoning for Human-Robot Interaction.* In IEEE/RSJ Int. Conf. on Intel. Rob. And Sys., San Diego, CA, USA, 2007. 21, 22, 26, 29, 30, 38, 40, 41, 68, 91

[Sisbot 08a] E. A. Sisbot, A. Clodic, R. Alami & M. Ransan. *Supervision and motion planning for a mobile manipulator interacting with humans.* In Proc. of HRI-2008, pages 327–334, Amsterdam, Netherlands, 2008. ACM/IEEE. 99

[Sisbot 08b] Emrah Akin Sisbot. *Towards human-aware robot motions.* PhD thesis, Université Paul Sabatier, Toulouse, 2008. 2

[Sisbot 10] E.A. Sisbot, L.F. Marin-Urias, X. Broquère, D. Sidobre & R. Alami. *Synthesizing Robot Motions Adapted to Human Presence*. International Journal of Social Robotics, 2010. 26, 28, 41

[Sisbot 12] E.A. Sisbot & R. Alami. *A Human-Aware Manipulation Planner*. 2012. 26, 29, 40, 42

[Stentz 95] A. Stentz & M. Hebert. *A complete navigation system for goal acquisition in unknown environments*. Autonomous Robots, vol. 2, no. 2, pages 127–145, 1995. 14

[Svenstrup 10] M. Svenstrup, T. Bak & H.J. Andersen. *Trajectory planning for robots in dynamic human environments*. In Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on, pages 4293–4298. IEEE, 2010. 23

[Takemura 07] H. Takemura, K. Ito & H. Mizoguchi. *Person following mobile robot under varying illumination based on distance and color information*. In Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on, pages 1500–1505. IEEE, 2007. 21, 25

[Theodorou 10] E. Theodorou, J. Buchli & S. Schaal. *Reinforcement learning of motor skills in high dimensions : A path integral approach*. In Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 2397–2403. IEEE, 2010. 36

[Tinbergen 51] N. Tinbergen. The study of instinct. Oxford University Press, London and New York, 1951. 100

[Trafton 05] J. Gregory Trafton, Nicholas L. Cassimatis, Magdalena D. Bugajska, Derek P. Brock, Farilee E. Mintz & Alan C. Schultz. *Enabling effective human-robot interaction using perspective-taking in robots*. IEEE Transactions on Systems, Man, and Cybernetics, vol. 35, pages 460–470, 2005. 99

[Traver 00] VJ Traver, AP del Pobil & M. Perez-Francisco. *Making service robots human-safe*. In Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on, volume 1, pages 696–701. IEEE, 2000. 21

[Urmson 03] C. Urmson & R. Simmons. *Approaches for heuristically biasing RRT growth*. 2003. 15

[Švestka 97] P. Švestka. *Robot Motion Planning using Probabilistic Roadmaps*. PhD thesis, Universiteit Utrecht, 1997. 1, 8, 9

[Walters 05a] M.L. Walters, K. Dautenhahn, K.L. Koay, C. Kaouri, R. Boekhorst, C. Nehaniv, I. Werry & D. Lee. *Close encounters : Spatial distances between people and a robot of mechanistic appearance*. In Humanoid Robots,

2005 5th IEEE-RAS International Conference on, pages 450–455. IEEE, 2005. 19

[Walters 05b] M.L. Walters, K. Dautenhahn, R. Te Boekhorst, K.L. Koay, C. Kaouri, S. Woods, C. Nehaniv, D. Lee & I. Werry. *The influence of subjects' personality traits on personal spatial zones in a human-robot interaction experiment.* In Robot and Human Interactive Communication, 2005. RO-MAN 2005. IEEE International Workshop on, pages 347–352. IEEE, 2005. 19

[Wedge 08] N.A. Wedge & M.S. Branicky. *On heavy-tailed runtimes and restarts in rapidly-exploring random trees.* In Proceedings of the First Symposium on Search Techniques in Artificial Intelligence and Robotics, Chicago, Illinois, USA, pages 127–133, 2008. 15

[Wilmarth 98] S.A. Wilmarth, N.M. Amato & P.F. Stiller. *MAPRM : A Probabilistic Roadmap Planner with Sampling on the Medial Axis of the Free Space.* In IEEE Int. Conf. Robot. And Autom., pages 1024–1031, 1998. 10

[Yamaoka 08] Fumitaka Yamaoka, Takayuki Kanda, Hiroshi Ishiguro & Norihiro Hagita. *How close ? : model of proximity control for information-presenting robots.* In ACM/IEEE International Conference on Human-Robot Interaction, HRI, pages 137–144, Amsterdam, The Netherlands, 2008. 19

[Yoda 95] M. Yoda & Y. Shiota. *Basic study on avoidance motions for human behaviors.* In Robot and Human Communication, 1995. RO-MAN'95 TOKYO, Proceedings., 4th IEEE International Workshop on, pages 318–322. IEEE, 1995. 19

[Yoda 97] M. Yoda & Y. Shiota. *The mobile robot which passes a man.* In Robot and Human Communication, 1997. RO-MAN'97. Proceedings., 6th IEEE International Workshop on, pages 112–117. IEEE, 1997. 21, 24

[Yoshimi 06] T. Yoshimi, M. Nishiyama, T. Sonoura, H. Nakamoto, S. Tokura, H. Sato, F. Ozaki, N. Matsuhira & H. Mizoguchi. *Development of a person following robot with vision based target detection.* In Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, pages 5286–5291. IEEE, 2006. 21, 25

[Zinn 04] M. Zinn, O. Khatib, B. Roth & J. K. Salisbury. *Playing it safe [human-friendly robots].* IEEE Robotics & Automation Magazine, 2004. 18

[Zucker 07] M. Zucker, J. Kuffner & M. Branicky. *Multipartite rrts for rapid replanning in dynamic environments.* In Robotics and Automation, 2007 IEEE International Conference on, 2007. 14, 90